PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 5 : G06F 15/22

A1

(11) International Publication Number: WO 94/01826

(43) International Publication Date: 20 January 1994 (20.01.94)

(21) International Application Number: PCT/US93/06027

(22) International Filing Date: 23 June 1993 (23.06.93)

(30) Priority data:
908,310 1 July 1992 (01.07.92) US

(71) Applicant: MASPAR COMPUTER CORPORATION [US/US]; 749 North Mary Avenue, Sunnyvale, CA 94086 (US).

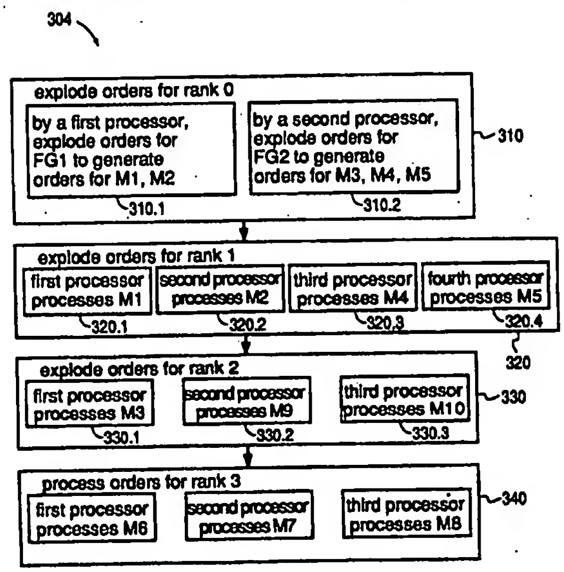
(72) Inventors: BROWN, John, S. ; 366 Warec Way, Las Altos, CA 94022 (US). BLANK, William, T. ; 1501 Stanford Avenue, Palo Alto, CA 94306 (US). HOLT, Mark, W. ; 90 Old Mill Grove Road, Lake Zurich, IL 60047 (US).

(74) Agents: SHENKER, Michael et al.; Skjerven, Morrill, MacPherson, Franklin & Friel, 25 Metro Drive, Suite 700, San Jose, CA 95110 (US).

(81) Designated States: AU, CA, JP, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
With amended claims.

(54) Title: SYSTEMS AND METHODS FOR PLANNING AND SIMULATION



(57) Abstract

A fast system addressing the same problem as MRP-II is described which is suitable for a multi-processor computer. The system performs both material requirements planning and capacity resource planning. Material requirements planning is performed by the multi-processor computer (504) as follows. A process tree incorporating a bill of materials as well as routing information is ranked, and the orders for the materials of each rank are exploded before any order for any higher rank material is exploded. Separate processors (536) explode in parallel the orders for separate materials of the same rank. Capacity Resource Planning is performed in parallel, a separate processor (536) planning the capacity of a separate work center. Simulation of material production is performed in parallel for each rank, separate processors simulating the production of separate materials of the same rank. The simulation is performed starting with the highest rank and proceeding in sequence to the lowest rank. Simulation of work center schedules is performed in parallel similarly to the MRP-II method described. The invention is applicable also to Just-in-Time manufacturing, and to planning and simulation in transportation industries.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	FR	France	MR	Mauritania
AU	Australia	GA	Gabon	MW	Malawi
BB	Barbados	GB	United Kingdom	NE	Niger
BE	Belgium	GN	Guinea	NL	Netherlands
BF	Burkina Faso	GR	Greece	NO	Norway
BG	Bulgaria	HU	Hungary	NZ	New Zealand
BJ	Benin	IE	Ireland	PL	Poland
BR	Brazil	IT	Italy	PT	Portugal
BY	Belarus	JP	Japan	RO	Romania
CA	Canada	KP	Democratic People's Republic of Korea	RU	Russian Federation
CF	Central African Republic	KR	Republic of Korea	SD	Sudan
CG	Congo	KZ	Kazakhstan	SE	Sweden
CH	Switzerland	LI	Liechtenstein	SI	Slovenia
CI	Côte d'Ivoire	LK	Sri Lanka	SK	Slovak Republic
CM	Cameroon	LU	Luxembourg	SN	Senegal
CN	China	LV	Latvia	TD	Chad
CS	Czechoslovakia	MC	Monaco	TG	Togo
CZ	Czech Republic	MG	Madagascar	UA	Ukraine
DE	Germany	ML	Mali	US	United States of America
DK	Denmark	MN	Mongolia	UZ	Uzbekistan
ES	Spain			VN	Viet Nam
FI	Finland				

SYSTEMS AND METHODS FOR PLANNING AND SIMULATION

This application includes a microfiche Appendix A. The total number of microfiche in Appendix A is 1. The total number of frames in Appendix A is 41.

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to planning and simulation by a computer, and more particularly to planning and simulation in manufacturing, transportation and other such areas by a computer.

Description of Related Art

In many areas including manufacturing, planning is sometimes done using a computer. In particular, computer planning is widespread in factories using MRP-type planning. MRP II, or Manufacturing Resource Planning II, is described in M. Baudin, Manufacturing Systems Analysis: with Application to Production Scheduling (Prentice Hall 1990), pp. 209-247. In MRP II-type factories, computer systems perform material requirements planning, that is, computer systems generate schedules containing material quantities and the due dates of production. Computer

systems also perform Capacity Requirements Planning (CRP), that is, computer systems generate schedules for work centers (such as machining, packaging, etc.) involved in production.

In factories processing tens of thousands of different materials at hundreds or thousands of work centers, planning may take a long time even when done by a computer system. It is desirable to provide computer systems that perform planning faster.

It is also desirable to provide fast computer systems for planning in other areas such as logistics planning in transportation. For example, it is desirable to provide a fast computer system for generating schedules for events in the airline industry including schedules for flight departures and arrivals, luggage check-in, schedules for fuel, food, personnel, gate resources, and so on.

SUMMARY OF THE INVENTION

The present invention provides a computer system for performing planning in manufacturing, transportation and other such areas. The computer system in some embodiments uses a multi-processor computer. The planning is separated into tasks performed in parallel by different processors. The separation into tasks is done so as to maximize the parallelism of the processor operation, that is, to maximize the proportion of work that the processors do in parallel. To that end, the tasks are structured to be largely independent from each other so as to reduce the time that the processors wait for one another during the task execution. The system speed is therefore increased.

The invention can be implemented efficiently in some embodiments on SIMD (Single Instruction Multiple Data) computers in which different processors execute at the same time the same instruction but possibly on different data.

These and other advantages are achieved in some systems using a rank-ordered tree technique. For example,

in manufacturing, the technique is used for material requirements planning. The tree representing the bill of materials is ranked. Backward explosion of the orders for the materials of the same rank is performed in parallel by different processors. A separate processor explodes the orders for a separate material. The explosion of an order for a material M involves generating the orders for input materials needed to produce material M. Since materials of the same rank are not input materials to one another, explosion of orders for one material of the rank does not generate orders for the other materials of the same rank. Hence, the processor exploding the orders for one material of the rank completes the explosion without waiting for another processor to complete the explosion of the orders for another material of the same rank. High speed is achieved as a result.

The present invention provides also a computer system for simulation. In manufacturing, the computer system simulates the production of materials given inventories and orders to outside suppliers. The rank-ordered tree technique is used in some embodiments so that the simulation of production of different materials of the same rank is performed in parallel by different processors. High speed is achieved in some embodiments.

The invention provides also planning and simulation systems for transportation. In some embodiments, a rank-ordered tree technique on a multi-processor computer is used to achieve high speed.

This summary does not purport to describe the invention in all generality. Other embodiments and variations of the invention are described below. The invention is defined by the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 diagrams an example of a relationship among materials and processes in a manufacturing facility.

Figure 2 diagrams a rank-ordered tree for the example of Figure 1.

Figure 3 is a flow chart of material requirements planning for the example of Figure 1.

Figure 4 is a flow chart of work center scheduling, including Capacity Requirements Planning, according to the invention.

Figure 5 is a block diagram of a multi-processor computer suitable for implementing the invention.

Figures 6A, 6B and 7 diagram data structures in the computer of Figure 5.

Figure 8 is a flow chart of manufacturing planning steps according to the invention.

Figure 9 is a flow chart showing in detail the step of creating a rank-ordered tree which step is shown generally in Figure 8.

Figures 10A, 10B and 11 are flow charts showing in detail the material requirements planning step of Figure 8.

Figure 12 is a data flow diagram for one step of Figure 11.

Figure 13 is a flow chart showing in detail the Capacity Requirements Planning step of Figure 8.

Figure 14 is a data flow diagram showing data flow in some steps of Figure 13.

Figure 15 is a flow chart showing in detail one step of Figure 8.

Figure 16 is a flow chart showing in detail an interprocessor data transfer technique used in some steps of Figure 8.

Figure 17 diagrams an example of a relationship between events in an airline industry.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 diagrams a relationship among materials and processes in a manufacturing facility in one example. FG1 and FG2 are finished goods. FG1 is manufactured by a

process P1 from materials M1 and M2. M1 and M2 are input materials to process P1, and FG1 is the output material. M1 is supplied by an outside supplier who is represented by a process P7 without input materials. M2 is manufactured by process P3 from material M3; M3 is manufactured by P4 from M6, M7 and M8; FG2 is manufactured by P2 from M3, M4 and M5, and so on. Materials M4, M6, M7, M8 and M10 are purchased parts, i.e. they are supplied by an outside supplier (or suppliers).

Each of processes P1 through P6, which have input materials, is performed by a work center (not shown) on a manufacturing floor. Different processes may or may not be performed by the same work center.

Process P1 is characterized by a setup time and a run time per piece of output material, as shown at 110. The time needed to manufacture N pieces of FG1 by process P1 is:

$$\text{setup\_time} + N*\text{run\_time\_per\_piece}. \qquad (1)$$

P7 is similarly characterized by its own setup time and run time per piece, as shown at 120. For P7, the expression (1) above describes the time that it takes the supplier represented by P7 to provide N pieces of M1. Every process P1 through P12 is characterized by a setup time and a run time per piece.

Figure 1 combines material information commonly represented by a bill of materials, along with process information. For a description of the bill of materials, see M. Baudin, Manufacturing Systems Analysis: with Application to Production Scheduling (Prentice Hall, 1990), incorporated by reference herein, pages 93-94. As is seen in Figure 1, there exists a tree relationship among materials. The tree relationship is as follows: for each process P having input materials, the output material of P is a parent of the input materials, and the input materials are children of the output material. For

example, FG1 is a parent of children M1 and M2, M2 is a parent of child M3, and so on.

As shown at 130, child material M1 is characterized by "qty per parent"--the quantity of M1 needed to produce one piece of the parent material FG1. Every material M1 through M10 is characterized similarly by a "qty per parent". Material M3 has two parents--M2 and FG2--and M3 may have different quantities per parent for M2 and FG2. M8 also has two parents, and M8 may also have different "qty per parent" parameters for its different parents.

In addition to representing a material, each symbol FG1, FG2, and M1 through M10 represents an event that the corresponding material has been produced. The planning systems described below with reference to Figures 3-16, plan these events generating material requirements schedules and work center schedules.

The manufacturing facility receives, from customers, orders for finished goods FG1 and FG2 such as shown at 140 for FG1. Each order is characterized by a quantity and a due date. An order for FG1 generates an order for M1 (such as order 150) and an order for M2. Generating the orders for the input materials M1 and M2 from the orders for the output material FG1 is termed "backward explosion".

Similarly, the orders for M2 are exploded to generate orders for M3, the orders for FG2 are exploded to generate orders for M3, M4 and M5, and so on. Propagating the orders for finished goods through the remaining materials is termed "netting of the materials requirements". The material requirements planning schedule, i.e. the list of production quantities and due dates, is generated as a result. Each order is a schedule entry in the material requirements planning schedule.

The orders are generated using "setup time" and "run time per piece" of the processes P and using "quantity per parent" of the input materials. For example, when order 140 is exploded to generate order 150 for M1, the "qty" of

order 150 is determined in some embodiments by multiplying the "qty" of order 140 by "qty per parent" at 130. The "due date" of order 150 is determined in some embodiments as: ("due date" of order 140) minus (the time T that it takes to manufacture the "qty", as defined by order 140, of material FG1), where T is determined from the "setup time" and "run time per piece" at 110.

The material requirements planning schedule generation is performed on a multi-processor computer. In order to maximize the processor utilization and hence the execution speed, the schedule generation is performed so as to maximize the parallelism of the processor operation, that is, to maximize the proportion of work that different processors do in parallel. To that end, the tasks for each processor are structured to be largely independent from each other so as to reduce the time that the processors wait for one another during the task execution.

To maximize parallelism, a rank-ordered tree technique is used as illustrated in Figure 2. Each material is assigned a rank which is the maximum level of the material in the tree of Figure 1. Materials FG1 and FG2 each have level 0 and hence rank 0. The children of each material in the tree have the level one greater than the parent material. Thus materials M1, M2, M4 and M5 have level 1 and hence rank 1. Material M3 has level 2 as a child of M2 and level 1 as a child of FG2. Hence the rank of M3 is 2 (the maximum of 2 and 1). Materials M9 and M10 have rank 2, and materials M6, M7 and M8 have rank 3, as illustrated in Figure 2.

During material requirements planning, the orders for separate materials of the same rank are exploded by separate processors as is illustrated in Figure 3 at 304. The orders are exploded starting with the lowest rank 0 and proceeding in sequence to the highest rank but one, i.e. to rank 2. At step 310, the planning system explodes the orders for the rank 0 materials FG1 and FG2, generating orders for M1 through M5. One processor

("first processor") explodes the orders for FG1 (step 310.1) while another ("second") processor, in parallel, explodes the orders for FG2 (step 310.2). Steps 310.1 and 310.2 are largely independent from each other. In particular, the explosion of the orders for FG1 (step 310.1) does not generate orders for FG2, and hence the second processor does not wait, at step 310.2, for the first processor to complete the explosion of the FG1 orders. Similarly, FG2 does not generate orders for FG1, and hence the first processor does not wait until the second processor completes the explosion at step 310.2. A high degree of parallelism, and hence high speed, are achieved as a result.

Step 310 is suitable for a highly parallel implementation on a SIMD computer. In such a computer, the first and second processors execute in parallel the same instructions, through possibly on different data. When an instruction must be executed by only one processor, the other processor is made inactive (disabled). A high degree of parallelism is achieved because the explosion of orders for FG1 and FG2 involves many of the same instructions and because, therefore, the first and second processors are both active during much of the explosion.

At step 320, the system explodes the orders for the rank 1 materials M2 and M5 as shown, respectively, at 320.2 and 320.4. Orders for M1 and M4 are not exploded since M1 and M4 do not have child materials--M1 and M4 are supplied by outside suppliers. However, certain processing is done for M1 and M4 at respective steps 320.1 and 320.3 to schedule the purchase of these materials from the suppliers and to track their inventory, as described more fully below. Steps 320.1, 320.2, 320.3 and 320.4 are performed in parallel, each by a separate processor.

The "first" and "second" processors of step 320 may or may not be the same as the "first" and "second" processors of step 310.

At step 330, in parallel, the rank 2 materials are processed. One processor explodes the orders for M3 (step 330.1); while another processor explodes the orders for M9 (step 330.2); while a third processor processes M10 (step 330.3). The "first", "second", and "third" processors of step 330 in Figure 3 need not be the same as the "first", "second", and "third" processors of steps 310 and 320.

At step 340, the rank 3 materials M6, M7 and M8 are processed in parallel each by a separate processor. The processors performing step 340 may or may not be the same as the processors performing steps 310, 320 and 330.

A high degree of parallelism is achieved at each step 310, 320, 330 and 340 because, at each step, processing of any material of the respective rank does not generate orders for another material of the same rank. A high degree of parallelism is achieved on a SIMD computer because processing of different materials of the same rank involves many of the same instructions.

At step 410 (Figure 4) the system performs Capacity Requirements Planning (CRP). For each process Pi, i = 1 through 12, a separate processor generates, at respective step 410.i, a schedule for the respective work center or the respective outside supplier. The processor reads information regarding the orders for the output material of the respective process (material FG1 for process P1, material M2 for process P3, and so on), and the processor generates the work center schedule, or the supplier schedule, needed to satisfy these orders. The processors operate in parallel. A high degree of parallelism is achieved because, given the orders for output materials, the schedule generation tasks for different processes Pi are largely independent from each other even if the processes Pi are to be performed by the same work center. A high degree of parallelism is achieved on a SIMD computer because much of the schedule generation involves execution of the same instructions for each process Pi.

In some embodiments, each work center schedule is managed by a separate processor. In such embodiments, at step 410.1, the first processor collects the order information for material FG1 and communicates that information to the processor managing the schedule of the work center of process P1, say, processor PP1. Processor PP1 updates the work center schedule accordingly. Similarly, at step 410.2, the second processor collects the order information for FG2 and communicates that information to the processor managing the schedule for the work center of P2, say, processor PP2. The first and second processors operates in parallel with each other, and processors PP1 and PP2 also operate in parallel with each other.

Step 410.3 and other such step are performed in the same way in these embodiments.

Optionally, at step 420, the system matches each work center schedule against the work center available capacity to determine if the work center is overloaded. Step 420 is performed in parallel by a separate processor for each work center. If at step 420 a work center is found overloaded, the planner can relieve the overload by, for example, changing a due date (e.g., renegotiating a due date with a customer), adding capacity (e.g., adding machinery or a work shift), expediting a scheduled receipt of inventory parts, or generating a firm planned order to shift some work to an earlier time period. The planner can then reschedule the production by repeating steps 304 and 410. A high degree of parallelism is achieved at step 420 because overload checking operations for different work centers are largely independent from each other. A high degree of parallelism is achieved on a SIMD computer because much of overload checking involves the same instructions for each work center.

As described more fully below with respect to Figures 5-16, the planning system of Figures 3 and 4 is efficiently implemented on computers having thousands of

processors. On such computers, the system performs planning quickly even for factories producing tens of thousands of materials on thousands of work centers.

Figures 5-16 show in detail an implementation of the planning system illustrated by Figures 3, 4. The planning system of Figures 5-16 is implemented on a SIMD computer 504 (Figure 5) of type MP-1™ available from the MasPar Computer Corporation of Sunnyvale, California. Computers of type MP-1™ have at least 1024 processors. These computers are described in the following manuals which are available from the MasPar Corporation and are hereby incorporated herein by reference: <u>MasPar System Overview</u>, <u>MPPE User Guide</u>, and <u>MasPar Commands</u>. These manuals are packaged in a package having Part Number 9300-9001-00. The package contains also the following two manuals describing a C-like programming language MPL™ (MasPar Parallel Application Language) which is suitable for programming the MP-1 type computers: <u>MPL User Guide</u> and <u>MPL Reference Manual</u>. These two manuals are also incorporated herein by reference.

Computer 504 contains a front end 510 and a data parallel unit ("DPU") 520. Front end 510 is a one-processor system containing a processor 522, a memory 524, a computer terminal 526, and other I/O devices (not shown) such as a disk drive.

DPU 520 contains an array 530 of at least 1024 processor elements (PEs) such as PEs 532.1, 532.2, 532.3, 532.4 and others (not shown). Each PE 532.i has a processor, such as processor 536, and a memory accessible to the processor, such as memory 538.

DPU 520 includes an array control unit ("ACU") 542 that controls PE array 530. ACU 542 includes processor 544 and memory 546.

DPU 520 includes also a router 550 for moving data between the memories 538 of different PEs 532.

Computer 504 is a SIMD (Single Instruction Multiple Data) computer. All the active (<u>i.e.</u>, enabled) PEs 532

execute in parallel the same instruction broadcast by the ACU 542. Each PE 532 accesses its own memory 538 when executing certain ACU instructions.

The software source code for the planning system is reproduced in Appendix A. The source code planning modules are written in MPL.

Figures 6A, 6B illustrate some data structures in PE memories 538. Each process P such as processes P1 through P12 of Figure 1 is represented by an operation record such as records 610.1 and 610.2. The operation records 610 are distributed among PEs 532 evenly so that each PE 532 contains the same number of operation records 610, except that some PEs 532 may contain an extra operation record 610 to accommodate excess records. If the number of processes P is smaller than the number of PEs 532, some PEs 532 contain no operation records.

In each PE 532, its operation records 610 are arranged in an array. In each PE 532, variable "P_op_head_ptr" points to the beginning of the array. (According to the variable naming convention adopted herein and in Appendix A, prefix "p_" indicates that the variable is plural, <u>i.e.</u>, it is reproduced in every PE 532. Prefix "s_" indicates that the variable is singular, <u>i.e.</u>, it is defined in memory 546 of ACU 542.)

Each operation record 610 has a field "output_id" identifying the output material of the corresponding process P. For example, if operation record 610.1 represents process P1 of Figure 1, the "output_id" of the record will identify material FG1.

Each operation record 610 has a pointer "input_ptr" which points to a list of records such as records 620.1 and 620.2. Each record 620 represents an input material of the respective process. In one example in which the operation record 610.1 represents process P1, record 620.1 represents material M1 and record 620.2 represents material M2. For "supplier" processes which have no input

materials (such as process P7 in Figure 1), the "input_ptr" field is NULL.

Each operation record 610 has a pointer "curr_ds_ptr" pointing to a possibly empty list of demand/supply records such as records 630.1 and 630.2. Each demand/supply record 630.i represents an order for the material "output_id" or an inventory of the material. Each demand/supply record 630 has a "quantity" field and a "due_date" field. If quantity > 0, record 630 represents an inventory available at "due_date". If quantity < 0, record 630 represents an order for "-quantity" pieces to be provided on "due_date".

Field "wc_mgr_pe" of operation records 610 will be described below.

As seen in Figure 1, each material is an output of some process. In particular, the material corresponding to a record 620 is an output of a process P represented by some operation record 610.i. Field "op_mgr_pe" of record 620 identifies the PE 532 containing the operation record 610.i, and field "op_link" identifies the location of the operation record 610.i in the operation record array of that PE. For example, if record 620.1 represents material M1 of Figure 1, then: "op_mgr_pe" of record 620.1 identifies the PE 532 containing the operation record 610.i representing process P7, and "op_link" identifies the position of the operation record 610.i in the operation record array of that PE 532. That PE 532 may or may not be the same PE as the PE containing the record 620.1.

Each record 620 contains "quantity_per_parent" described above in connection with Figure 1.

Each work center is described, in some PE 532, by data 640 (Figure 6B). Capacity record 650 (plural "P_wc_capacity") contains the work center identification "wc_id" and the maximum work center capacity normally available--"normal_capacity". The work center capacity in this embodiment is measured as the number of processes P

that execute on the work center simultaneously at a given time. "normal_capacity" is the maximum number of processes P that normally can execute on the work center simultaneously.

Pointer "except_ptr" in record 650 points to a list, possibly empty, of exception-capacity records such as records 656.1 and 656.2. Each exception record 656 stores an altered maximum capacity of the work center (field "altered_capacity") and the times "begin_on" and "end_on" between which the "altered_capacity" maximum applies.

During CRP step 410 (Figure 4), PE 532 creates in its memory 538 a list of work center schedule records, such as records 660.1 and 660.2, for the work center represented by the capacity record 650 in the same PE. Plural variable "P_wc_sched_head" points to the beginning of the list of records 660. Each record 660 has a field "material_id" identifying the output material to be produced during the time period defined by fields "start_at" and "end_on" in the record. The "material_id" is the "output_id" of the operation record 610 representing the process P performed at the work center between the times "start_at" and "end_on". Each record 660 contains also the quantity to be produced ("qty_sched") and further contains the "setup_time" and "run_time_per_piece" parameters from the corresponding operation record 610.

Each operation record 610 has a field "wc_mgr_pe" identifying the PE which has the data 640 for the corresponding work center.

In the embodiment of Figures 5-16, the total number of work centers does not exceed the total number of PEs 532. Each work center is described by data 640 in exactly one PE 532. If the number of work centers is less than the number of PEs 532, some PEs 532 will not have work center data.

In some other embodiments, the total number of work centers exceeds the total number of PEs 532.

Figure 7 shows the format of the operation records 610 more fully. Each operation record 610 has fields "curr_rank", "next_rank", "curr_state" and "next_state" which are used for determining the rank of the respective output material "output_id" as described below. Operation record 610 also has fields "setup_time" and "run_time_per_piece" for the respective process P. These fields contain the parameters shown at 110 and 120 in Figure 1.

The fields "lot_minimum" and "lot_multiple" of record 610 define the possible lot sizes of the output material produced by respective process P. The lot size is at least "lot_minimum", and the lot size is a multiple of "lot_multiple".

The field "init_ds_ptr" is a pointer to a list of records 710.i representing initial inventory and some other data as described below.

Figure 8 is a flow chart showing a sequence of planning steps performed using the planning system. At step 810, the planning system reads in the following data: (1) the orders for finished goods (FG1 and FG2 in Figure 1), and (2) inventory information. These data are placed at step 810 on the "init_ds_ptr" list of the respective operation records 610 as is illustrated in Figure 7. In the example of Figure 7, the operation record 610 corresponds to process P1 of Figure 1. The system has read an order for five (5) pieces of output material FG1. Accordingly, the system creates a record 710.1 and sets its "quantity" to -5. The system has read also inventory information indicating that three (3) pieces of material FG1 will be in the inventory on a certain due date. The system creates record 710.2 on the "init_ds_ptr" list and sets its "quantity" to 3.

At step 820, the system determines the ranks of the materials (<u>see</u> Figure 2). Next at step 304, the system explodes the orders to generate the material requirements planning schedule (<u>see</u> Figure 3). At step 410, the system

performs Capacity Requirements Planning to determine the work center schedules (see Figure 4). At step 420, the system checks the work center schedule for overloading (see Figure 4). Step 420 is optional in some embodiments. At step 830, the planner may modify the data on orders, inventory and capacity so as, for example, to relieve an overload or to update the data. For example, to relieve an overload, the planner may enter a firm planned order to produce some material in advance. Control then passes to step 820 to redo the planning.

In some embodiments, control from step 830 passes to step 304 rather than to step 820.

Figure 9 shows a flow chart of step 820. Step 820 corresponds to procedure "rank_tree" in Appendix A. At the completion of step 820, the "curr_rank" field of each operation record 610 (Figure 7) contains the rank of the respective output material. Step 820 is performed so as to maximize parallelism and to minimize interprocessor communication.

Step 820 of Figure 9 ranks only the materials participating in the explosion of orders. For example, if no orders has been read for material FG2 (Figure 1) at step 810 and no such orders has been created at step 830, material FG2 will not be ranked at step 820, and materials M4 and M5 will receive rank 0. Further, if no firm planned orders have been placed for material M5, M5 will not be ranked, and materials M9 and M10 will receive rank 0.

In some other embodiments, step 820 ranks all the materials.

At step 910, each operation record 610 is processed as follows: if the record's "init_ds_ptr" field has a record 710 with a negative "quantity" (indicating an order), "next_state" of the operation record 610 is set to TAGGED (a predetermined value) and "next_rank" is set to 0. Otherwise, "next_state" is set to UNTAGGED (a different value). The TAGGED materials (that is the

output materials corresponding to TAGGED operation records) will be ranked at subsequent steps of Figure 9 as described below. The UNTAGGED materials will not be ranked unless they will be TAGGED in a subsequent step of Figure 9.

Step 910 is a loop performed by PEs 532 in parallel. At the first iteration of the loop, the PEs 532 process their operation records 610 in the first positions in the respective operation record arrays (such as operation record 610.1 in Figure 6A). PEs 532 that have no operation records are inactive. The active PEs 532 process the records 610 as follows. First at step 910.1, at the same instructions from ACU 542 (Figure 5), the active PEs 532 set the "next_state" fields to UNTAGGED. Then at step 910.2 the active PEs 532 go in parallel through the respective "init_ds_ptr" records 710, comparing the "quantity" of each record 710 with 0. The PEs 532 process in parallel the first records 710.1 on the list, then in parallel the second records 710.2 on the list, and so on. When a PE 532 gets to the end of its list, it becomes inactive.

For each record 710, at the same instructions from ACU 542, the PEs 532 which find quantity < 0 set "next_state" of the respective operation record 610 to TAGGED and "next_rank" to 0, while the other PEs 532 are inactive.

At the second iteration of loop 910, PEs 532 process in the same way the operation records 610 in the second position in the operation records arrays (such as operation record 610.2 in Figure 6A). Those PEs 532 which have only one operation record 610 are inactive in the second iteration. At the third iteration, PEs 532 process similarly the operation records 610 in the third positions, and so on until all the operation records are processed.

At step 920, a singular variable "s_rank" is initialized to 0.

Step 930 is a loop executed while at least one operation record 610 has next_state = TAGGED. At step 930.1, "s_rank" is incremented. Step 930.2 is a loop performed by PEs 532, in parallel, once for the operation records 610 in each position in the operation record arrays. For each operation record 610, the respective PE 532, (1) copies "next_state" to "curr_state", and (2) sets "next_state" to UNTAGGED.

Step 930.3 is also a loop performed by PEs 532 in parallel once for each position in the operation record arrays. For each operation record 610, if its curr_state = TAGGED, the respective PE 532 performs step 930.3.1, and otherwise the PE 532 becomes inactive. Step 930.3.1 is a loop performed in parallel by the active PEs 532 once for each record 620 (Figure 6A) on the operation record's "input_ptr" list. As the PE 532 exhausts its "input_ptr" list, the PE 532 becomes inactive.

For each record 620, in the operation record 610 corresponding to the material represented by record 620, "next_rank" is set to "s_rank" and "next_state" is set to TAGGED at step 930.3.1.1. For example, suppose that in Figure 6A record 620.1 represents material M1 of Figure 1. In this case, fields "op_mgr_pe" and "op_link" of record 620.1 identify the operation record 610 of process P7. If step 930.3.1.1 is performed for record 620.1, then in the operation record 610 of process P7 "next_rank" is set to "s_rank" and "next_state" is set to TAGGED. The operation record 610 of process P7 may be located in a PE 532 which has been made inactive. For example, that PE 532 may be inactive because it has exhausted its "input_ptr" list. That PE 532 is then made active to set the "next_rank" and "next_state" of the record 610 as described above. The "next_rank" and "next_state" are set by the respective PEs 532 in parallel.

It can be seen that at the end of procedure 820, the "curr_rank" of each operation record 610 which

participated in ranking is set to the rank of the "output_id" material of the record.

Figures 10A and 10B show a flow chart of the MPS generation step 304. Step 304 corresponds to procedure "schedule" in Appendix A.

At step 1010, initialization is performed. Step 1010 is a loop performed by PEs 532 in parallel for each position in the operation record arrays. For each operation record 610, the respective PE 532 reinitializes the "curr_ds_ptr" list of the record. Namely, at step 1010.1, the linked list of demand/supply records 630 which is pointed to by "curr_ds_ptr" is deleted. At step 1010.2, the "init_ds_ptr" records 710 of each operation record 610 are copied to the "curr_ds_ptr" list. Thus, at the conclusion of step 1010, the records 630 on the "curr_ds_ptr" lists contain the initial information on orders and inventory for the respective output material.

At step 1020 (Figure 10B), the orders are exploded. Step 1020 is a loop performed once for each rank. Thus each step 310, 320, 330 and 340 of Figure 3 is performed as one iteration of loop 1020.

The loop variable "s_rank" defined in ACU 542 (Figure 5) ranges from 0 to the maximum rank.

Each iteration of loop 1020 performs step 1020.1. Step 1020.1 is itself a loop performed by PEs 532 in parallel once for each position in the operation record arrays. For each operation record 610, the respective PE 532 compares the operation record's "curr_rank" with "s_rank". If curr_rank ≠ s_rank, the PE becomes inactive. Otherwise the PE performs steps 1020.1.1, 1020.1.2 and 1020.1.3.

At step 1020.1.1, each active PE 532 sorts the "curr_ds_ptr" records 630 of the respective operation record 610 in the order of increasing "due_date". If more than one demand/supply records 630 on the list have the same "due_date", they are sorted so that the inventory

(i.e. supply) records (with quantity > 0) precede the order (i.e. demand) records (with quantity < 0).

At step 1020.1.2 a plural variable "p_QOH" (plural Quantity On Hand) is initialized to 0 in parallel by all active PEs 532. "p_QOH" will keep track of the inventory.

Step 1020.1.3 is a loop performed by PEs 532 in parallel once for each demand/supply record 630 on the respective "curr_ds_ptr" list. PEs 532 start from the beginning of the respective lists (and hence at the earliest due date) and proceed in order to the end (and hence to the latest due date). For each demand/supply record 630, the respective PE 532 at step 1020.1.3.1 explodes the order represented by record 630 if record 630 represents an order, or does other processing as needed, as explained below in connection with Figures 11 and 12. Once a PE exhausts its "curr_ds_ptr" list, the PE becomes inactive.

Figure 11 shows a flow chart of step 1020.1.3.1. At step 1110, each active PE 532 determines whether the respective demand/supply record 630 represents a firm planned order. In general, record 630 may represent: (1) a firm planned order; (2) an order other than a firm planned order; or (3) inventory. The records 630 which represent firm planned orders are recognized by a special field, not shown. If the record 630 does not represent a firm planned order, the PE sets:

$$p\_QOH = p\_QOH + (\text{"quantity" of record 630}) \qquad (2)$$

to reflect the change in inventory. Namely, if record 630 represents inventory, then quantity > 0, and "p_QOH" is increased by the operation (2) to reflect an addition to the inventory. If record 630 represents a non-firm-planned order, then quantity < 0, and "p_QOH" is decreased at (2) to reflect a reduction of the inventory. The inventory is reduced to satisfy the order.

If record 630 represents a firm planned order, "p_QOH" is not changed.

At step 1120, if record 630 represents an order that cannot be satisfied from inventory, the order is exploded. If record 630 represents inventory, or if record 630 represents an order that can be satisfied from inventory, the respective PE 532 becomes inactive. The available inventory is the value of "p_QOH" immediately before step 1110. The PE 532 determines whether the order can be satisfied from inventory, by comparing the absolute value of the "quantity" of record 630 with the value that "p_QOH" had immediately before step 1110.

The orders are exploded in parallel using steps 1120.1-1120.4. At step 1120.1, PEs 532 set the plural variable "p_prod_qty" to the quantity to be produced. The PE 532 determines "p_prod_qty" as the difference between (1) the absolute value of the "quantity" of record 630 and (2) the value of "p_QOH" immediately before step 1110. "p_prod_qty" is then increased, if necessary, to the nearest number which is at least "lot_minimum" and which is a multiple of "lot_multiple", where "lot_minimum" and "lot_multiple" are taken from the respective operation record 610.

At step 1120.2, each active PE 532 sets the plural variable "p_need_date" to the time at which the respective process P must start production to satisfy the order represented by record 630. "p_need_date" is determined as follows:

p_need_date = ("due_date" of record 630) minus (production time)

where

production time = setup_time + p_prod_qty * run_time_per_piece

and where "setup_time" and "run_time_per_piece" are provided by the operation record 610 (Figure 7). In some embodiments, different time units are used for "due_date" in demand/supply records 630 on the one hand and for "setup_time" and "run_time_per_piece" in the operation records 610 on the other. Any necessary time unit conversion is performed so that "p_need_date" is determined in the same time units as "due_date".

At step 1120.3, the orders are exploded to generate orders for respective child materials as illustrated in Figure 12. In Figure 12, PE 532.1 explodes the order represented by demand/supply record 630.1. Record 630.1 is on the "curr_ds_ptr" list of operation record 610.1. PE 532.1 goes through the records 620 on the "input_ptr" list of the record 610.1, in parallel with other PEs. Step 1120.3 is a loop executed once for each "input_ptr" record. For each record 620 on the list:

(1) The operation record 610.2 is found which corresponds to the input material of record 620. Record 610.2 is found using the fields "op_mgr_pe" and "op_link" of record 620. In Figure 12, "op_mgr_pe" identifies PE 532.2.

(2) A new demand/supply record 630.2 is added to the "curr_ds_ptr" list of operation record 610.2. The "due_date" and "quantity" of record 630.2 are set as follows:

due_date = "p_need_date" from PE 532.1; (3)

quantity = -(p_prod_qty * quantity_per_parent) (4)

where:

"p_prod_qty" is from PE 532.1, "quantity_per_parent" is from record 620.

If PE 532.2 was inactive, it is made active to create record 630.2. The right hand side values in (3) and (4) are moved from the PE 532.1 to PE 532.2 using the technique described below in connection with Figure 16.

At step 1120.4 (Figure 11), PE 532.1 creates a scheduled work record 1210, shown in Figure 12, to represent scheduled work for the respective work center. In scheduled work record 1210,

quantity = p_prod_qty;

due_date = "due_date" of record 630.1.

Scheduled work record 1210 is inserted in the "curr_ds_ptr" list of operation record 610.1 immediately after the demand/supply record 630.1. Scheduled work records 1210 are used at the CRP step 410 (Figure 8) as described below.

Step 1120.4 is performed even if the "input_ptr" list of the operation record 610.1 is empty, that is, even if the operation record represents an outside supplier. Scheduled work record 1210 then will be used to schedule the purchase of the respective material from the supplier.

At step 1130, "p_QOH" is incremented by "p_prod_qty" to reflect the inventory increase due to the production of the "p_prod_qty" pieces. "p_prod_qty" was initialized to 0 before step 1120. Hence, if steps 1120.1 through 1120.4 were skipped (as they are for records 630 representing inventory and for records 630 representing orders that can be satisfied from inventory), "p_QOH" is unchanged by step 1130.

As is illustrated by Figure 10B, the orders for separate materials of the same rank are exploded, in general, by separate PEs 532 though in some cases different operation records 610 in the same PE 532 have the same rank and hence the orders for the respective

"output_id" materials are exploded in those cases by the same PE 532 consecutively.

Figure 13 shows a flow chart of CRP step 410 (Figure 8). At step 1308, the PEs 532 in parallel empty their respective "P_wc_sched_head" lists. At step 1310, the system creates work center schedule records 660 (Figure 6B). Step 1310 is a loop performed by PEs 532 in parallel once for each position in the operation record arrays. Each iteration 1310.1 of the loop is itself a loop performed once for each record on the "curr_ds_ptr" list of the respective operation record 610. The records on the "curr_ds_ptr" list are examined starting from the beginning of the list. If the record being examined is a demand/supply record 630, the respective PE 532 becomes inactive. If the record is a scheduled work record 1210 (Figure 12), steps 1310.1.1 and 1310.1.2 are performed as illustrated in Figure 14. Figure 14 shows records 610 and 1210 in a PE 532.1. At step 1310.1.1, PE 532.1 creates, in its own memory 538, a temporary record 1410 having the same format as records 660 and having the following field values:

(i) setup_time, run_time_per_piece are from the operation record 610.

(ii) material_id = output_id from the operation record 610.

(iii) end_on = due_date from record 1210.

(iv) start_at = end_on - (setup_time + quantity*run_time_per_piece) where "quantity" is provided by record 1210. Time units conversion is performed as needed.

(v) qty_sched = "quantity" from record 1210.

At step 1310.1.2, temporary record 1410 is copied to a record 660 in the PE 532.2 identified by the "wc_mgr_pe" of operation record 610. PE 532.2 adds record 660 to the respective "P_wc_sched_head" list.

At step 1320, PEs 532, including PE 532.2 in Figure 14, sort in parallel their respective

"P_wc_sched_head" lists of records 660 in the order of the increasing "start_at" times. This concludes the CRP step 410.

Step 420 (Figure 8), optional in some embodiments, is performed to determine whether any work center is overloaded, that is, whether the work center scheduled capacity, as determined by the respective records 660, exceeds the maximum capacity as determined by the respective capacity record 650 and exception records 656. Step 420, illustrated in Figure 15, is performed as follows.

The time in the planning system is represented by discrete time points. Each field "start_at" and "end_on" of work center schedule records 660 corresponds to one such discrete time point. Each PE 532 has in its memory an array "P_wc_load_profile". This array has, in each PE 532, one location for each discrete time point "t" to store a work center capacity.

At step 1510 in Figure 15, PEs 532, in parallel, set P_wc_load_profile[t], for all points "t" in the planning period, as follows:

> P_wc_load_profile[t] = scheduled capacity, at time "t", of the work center corresponding to the PE 532.

The PEs that do not correspond to a work center are inactive. The active PEs 532 start at the beginning of their respective "P_wc_sched_head" lists and proceed in order to the end. For each record 660 on the list, the respective PE 532 increments by one all locations P_wc_load_profile[t] for which:

$$\text{start\_at} \le t < \text{end\_on},$$

wherein "start_at" and "end_on" are from record 660. P_wc_load_profile[t] was initialized to 0 for all "t"

immediately before step 1510. As a result, at the conclusion of step 1510 P_wc_load_profile[t] = the number of all records 660 in the respective "P_wc_sched_head" list which have start_at ≤ t < end_on. This number represents, in this embodiment, the scheduled capacity of the respective work center.

At step 1520, for each time "t", the available capacity of each work center is determined and compared with the scheduled capacity P_wc_load_profile[t]. More particularly, at step 1520.1, PEs 532 examine in parallel their respective exception records 656 (Figure 6B). If PE 532 finds an exception record 656 whose fields "begin_on" and "end_on" satisfy:

$$\text{begin\_on} \le t < \text{end\_on}$$

then PE 532 sets a plural variable "p_capacity" to the "altered_capacity" of the exception record 656, and skips step 1520.2. Otherwise, at step 1520.2, PE 532 checks the singular array "S_shop_floor_calendar" to determine if the plant will be shut down at time t. For each time "t", S_shop_floor_calendar[t] is a flag showing whether the manufacturing facility is to be shut down at time "t". If the facility is to be shut down, PE 532 sets "p_capacity" to 0. Otherwise, "p_capacity" is set to "normal_capacity" of the respective capacity record 650. Hence at the conclusion of steps 1520.1 and 1520.2, "p_capacity" has the available capacity at time "t".

At step 1520.3, PEs 532 compare in parallel "p_capacity" with P_wc_load_profile[t]. If P_wc_load_profile[t] > p_capacity, an overload exists and a message is printed out to that effect. This completes step 420.

In the present embodiment, each outside supplier is represented, similarly to a work center, by a structure 640 (Figure 6B). In such a structure, the work center schedule records 660 represent a schedule for ordering the

materials from the supplier. "start_at" is the time the order must issue to the supplier. "end_on" is the time the ordered material is expected to arrive.

In the capacity record 650 corresponding to a supplier, "normal_capacity" is the maximum number of processes P, such as process P7 of Figure 1, that may have orders to the supplier outstanding at the same time. In the exception records 656, "altered_capacity" is the maximum number of such processes between "begin_on" and "end_on".

Steps 410 and 420 of Figures 13 and 15 are performed also on the structures 640 representing outside suppliers.

Figure 16 illustrates an interprocessor data transfer technique used in some steps of Figures 4-15 such as step 1120.3 of Figure 11 and step 1310.1.2 of Figure 13. In Figure 16 data are transferred by router 550 (Figure 5) in parallel from "source" PEs to "destination" PEs. Each PE 532 can be a source PE, a destination PE, both a source and a destination, or neither a source nor a destination. The PEs that are neither a source nor a destination are inactive during the transfer.

While router 550 can transfer data in parallel between different pairs of PEs, router 550 cannot, in some embodiments, transfer data is parallel from different source PEs to the same destination PE. The technique of Figure 16 serializes data transfers from multiple source PEs to the same destination PE while transferring the data in parallel to different destination PEs.

At step 1610 each source PE 532 sets a plural variable "p_have_inputs_to_send" to TRUE. Each source PE 532 will set its "p_have_inputs_to_send" to FALSE as the data from the PE are transferred to the respective destination PE 532.

Step 1620 is a loop performing the data transfer. At step 1620.1, the data are transferred only from source PEs 532 which have p_have_inputs_to_send = TRUE, though not necessarily from all such PEs. Each source PE 532 has a

variable "p_dest" identifying the respective destination PE. At step 1620.1, the source PEs, in parallel, call a function "connected (p_dest)" which returns a flag (TRUE or FALSE) indicating whether the calling PE can send data in parallel with other source PEs. If more than one source PEs, which call "connected," have the same "p_dest" value, "connected" will return TRUE to one of these PEs and FALSE to the other ones of these PEs. <u>See MPL Reference Manual</u>, <u>supra</u>. At steps 1620.1.1 and 1620.1.2, only those PEs send data to which "connected" has returned TRUE.

At step 1620.1.1, these PEs set their respective "p_have_inputs_to_send" to FALSE. At step 1620.1.2, the data from these PEs are transferred in parallel by router 550 to the respective destination PEs.

The rank-ordered tree technique is suitable also for performing computer simulation of manufacturing. Given initial inventories and material orders, the simulation indicates what other orders can be satisfied. For example, with reference to Figures 1 and 2, the simulation solves the following problem: given certain (possibly null) initial inventories for each material FG1, FG2 and M1 through M10, and given orders to outside suppliers for materials M1, M4, M6, M7, M8 and M10, determine what orders can be satisfied for the materials FG1, FG2, M2, M3, M5 and M9.

The simulation system processes the materials starting with the highest rank but one (<u>i.e.</u>, rank 2) and proceeding in sequence to the lowest rank 0. For each rank, the system determines the orders, <u>i.e.</u>, the quantities and production dates (due_date's), for each material of the rank. The system determines the orders from the following information: (1) orders for the materials of the higher ranks, and (2) inventories of the materials of the higher rank and the material whose orders are being determined.

One embodiment of the simulation system uses computer 540 (Figure 5) with the data structures of Figures 6A and 7. In performing the simulation for each given rank "s_rank", PEs 532 perform a loop executed in parallel once for each position in the operation record arrays. For each operation record 610 with curr_rank = s_rank, the respective PEs 532 determine in parallel the quantities and production dates for the respective "output_id" materials.

Once the orders are determined, the simulation system performs CRP step 410 to simulate the work center schedules. Optionally, the simulation system perform step 420 to check the work center schedules for overload.

The planning and simulation techniques of the invention are applied to Just-In-Time manufacturing facilities (JIT). In JIT, the planning system generates a schedule that defines, for each material, the rate of production of the material. Each order for a material contains the rate of production rather than quantity and due date. The planning system ranks the materials and explodes the orders rank by rank similarly to the MPS generation step 304 in the MRP case (Figures 3, 8, 10A, 10B). The orders for each rank are exploded in parallel, separate PEs 532 exploding the orders for separate materials. The work center schedules and overloads are also determined in parallel similarly to steps 410, 420 (Figures 4, 8, 13, 15) in the MRP case.

Simulation is performed for JIT systems similarly to the MRP case.

The rank-ordered tree technique is also used to plan the events for transportation industries such as airline industries. Figure 17 illustrates, for an airline, events related to a flight departure. "FLD" denotes a flight departure. FLD is an "output material", or rather an output event, of process P1 whose "input materials" or events are: M1--passengers are on board; M2--crew is on board and ready; and M3--aircraft is ready.

Event M1 is an output of process P2 whose inputs are: M4--boarding starts at the gate; and M5--luggage check-in starts at the ticket-luggage counter. M2 is an output of P3 whose input is M6--another flight manned by the same crew has arrived. M3 is an output of P4 whose inputs are: M7--start loading luggage into the aircraft; and M8--fuel truck arrives to fuel the aircraft. M7 is an output of P5 whose input is M5.

The events FLD and M1 through M8 of Figure 17 are ranked as explained above in connection with Figure 2. Each "order" in Figure 16 contains the time ("due date") by which the corresponding event must take place. The order may also have a "quantity". For example, the order for material M8 may include the quantity of fuel to be brought by the truck. An order may include a frequency with which an event must occur. For example, if an event consists in having loaded a fuel truck with aircraft fuel (this event is not shown in Figure 17), an order may include the number of trucks to be loaded per unit of time. Each order is an entry into a schedule of events, similarly to the Master Production Schedule in the MRP case.

The orders are exploded as in MPS generation step 304 to produce a schedule for each event.

A process P of Figure 17 can be associated with a "work center" having certain maximum capacity. For example, process P4 is associated with the "work center" which includes luggage-loading equipment, fuel trucks, fuel, and personnel. The "work center" scheduling is performed similarly to the CRP step 410 in the MRP case.

The "work center" schedule is then optionally checked for overloading as in step 420. If overloading exists, firm planned orders are used to change the schedule as in the MRP case. For example, in a system scheduling the flight departures for many aircraft, the "work center" schedule may require, at a given time, more fuel trucks to fuel the aircraft than are available. To relieve the

overload, the planner generates firm planned orders to fuel some aircraft at an earlier time.

Flight departures can be scheduled in parallel with other events such as arrivals.

Simulation in transportation systems is performed similarly to the MRP case.

The planning systems described above in connection with Figures 1-16 are used similarly in other transportation industries such as train, ship and truck industries. The systems are used also to schedule and simulate other movements of goods and people, such as movement of an army.

The embodiments described above are merely illustrative and are not intended to limit the scope of the invention. In particular, the invention is not limited by the type of the computer or the number of processors. Further, the invention is not limited by how the ranks are numbered in the rank-ordered trees. For example, with reference to Figure 2, the ranks are numbered in some embodiments from 1 to 4 rather than from 0 to 3. In other embodiments, the ranks are numbered in decreasing order, for example, from 3 to 0. In other words, materials FG1, FG2 are assigned rank 3, materials M1, M2, M4 and M5 are assigned rank 2, and so on. The explosion of orders starts from exploding the orders for materials FG1 and FG2, which materials are termed the materials "of the lowest rank" no matter how the ranks are numbered. The explosion is carried out in sequence shown in Figure 3 independently of how the ranks are numbered. Other embodiments and variations are within the scope of the invention as defined by the following claims.

CLAIMS

What is claimed is:

1. A method for determining a schedule for the manufacturing of one or more materials F, said method comprising the steps of:

determining one or more materials to be used in said manufacturing of said materials F;

defining a tree relationship among all said materials such that each material which has in the tree one or more child materials is to be produced using its child materials;

determining by a computer one or more orders for said materials F;

determining a rank in said tree of each said material including said materials F and said materials to be used, so that each material having one or more child materials has a lower rank than any one of its child materials;

for each rank R starting with the lowest rank and proceeding in sequence to the highest rank but one, if the rank R has at least one material MA which has one or more child materials C(MA) and for which there is at least one order O(MA), if said order O(MA) cannot be satisfied from inventory then exploding said order O(MA) to generate, by said computer, at least one order for at least one of said child materials C(MA).

2. The method of Claim 1 further comprising, before said exploding step, the step of determining, by said computer, one or more firm planned orders for at least one of said materials.

3. The method of Claim 1 wherein each order for a material MA comprises a quantity of the material and a due date by which that quantity is to be provided.

4. The method of Claim 1 wherein each order for a material MA comprises a rate at which the material is to be provided.

5. The method of Claim 1 wherein, for at least one rank R:

for one material MA1 of said rank R, exploding the order O(MA1) is performed by a processor PR1 of said computer; and

for another material MA2 of the same rank R, exploding the order O(MA2) is performed by another processor PR2 of said computer.

6. The method of Claim 5 wherein said processors PR1 and PR2 explode the respective orders O(MA1) and O(MA2) in parallel.

7. A method of scheduling the production of materials MA1 and MA2, materials MA1 and MA2 being such that neither material is used for producing the other material, by a computer having processors PR1 and PR2 which processors can operate in parallel, said method comprising the steps of:

entering into said computer one or more orders for said material MA1;

entering into said computer one or more orders for said material MA2; and

exploding, by each processor PRi, i=1,2, said orders for the respective material MAi to generate orders for one or more materials needed to produce the respective material MAi.

8. A method for scheduling interrelated events, said method comprising the steps of:

determining a tree relationship among said events such that for each event having one or more

child events in the tree, the child events must occur before the parent event;

ranking by a computer each event in said tree;

for one or more of said events, determining one or more entries in a schedule of said events, each entry being a schedule entry for one of said one or more events;

for each rank R starting from the lowest rank and proceeding in order to the highest rank but one, for each parent event E of the rank R, exploding by said computer one or more entries for the parent event E to generate one or more entries for each child event C(E) of said event E.

9. The method of Claim 8, wherein:

at least one rank R has at least two parent events E1 and E2; and

in said exploding step, one or more entries for E1 are exploded by a processor PR1 of said computer and one or more entries for E2 are exploded by a different processor PR2 of said computer.

10. The method of Claim 9 wherein said processors PR1 and PR2 explode the respective entries for E1 and E2 in parallel.

11. A manufacturing planning system comprising:

one or more processors;

a computer storage;

means for defining in said storage a tree relationship among materials such that each material having, in the tree, one or more child materials is to be produced using said child materials;

means for determining a rank of each said material in said tree so that any material having one or more child materials has a lower rank than any one of its child materials;

means for exploding, by said processors, the orders for all the parent materials of any given rank; and

means for invoking the exploding means for each rank in sequence from the lowest rank to the highest rank but one.

12. The system of Claim 11 wherein:

said processors are more than one in number; and

said exploding means comprises means for exploding two orders for two respective different materials by two different processors in parallel, each processor exploding one of said orders.

13. The system of Claim 12 further comprising a control unit for providing instructions to said processors, wherein each processor, when active, executes each instruction provided by said control unit.

14. An event planning system comprising:

a computer storage for storing information representing a tree of events;

means for determining a rank for each event in the tree so that any event that has a parent in said tree has a higher rank than the parent, and for storing said rank in said computer storage;

means for reading into said computer storage one or more schedule entries for one or more of said events; and

means for exploding schedule entries by one or more processors that can access said storage means, wherein said exploding means, when invoked:

examines parent events of each rank having a parent event, starting with the lowest rank and proceeding in sequence to the highest rank but one, to determine whether to explode any schedule entry for any parent event;

if the exploding means determines, in examining the parent events of any given rank R, that one or more entries ENT for at least one parent event of rank R are to be exploded, the exploding means explodes said one or more entries ENT before examining parent events of any rank higher than R; and

wherein each entry exploded by said exploding means is either (1) an entry read in by said reading means or (2) an entry generated by said exploding means when the exploding means explodes another entry.

AMENDED CLAIMS
[received by the International Bureau on 27 October 1993 (27.10.93); original claims 1,8,11 and 14 amended; new claims 15-53 added; other claims unchanged (11 pages)]

What is claimed is:

1. A method for determining a schedule for the manufacturing of one or more materials F, said method comprising the steps of:

determining one or more materials to be used in said manufacturing of said materials F;

defining a tree relationship among all said materials such that each material which has in the tree one or more child materials is to be produced using its child materials;

determining by a computer one or more orders for said materials F; and

for each rank R in said tree starting with the lowest rank and proceeding in sequence to the highest rank but one, if the rank R has at least one material MA which has one or more child materials C(MA) and for which there is at least one order O(MA), if said order O(MA) cannot be satisfied from inventory then exploding said order O(MA) to generate, by said computer, at least one order for at least one of said child materials C(MA).

2. The method of Claim 1 further comprising, before said exploding step, the step of determining, by said computer, one or more firm planned orders for at least one of said materials.

3. The method of Claim 1 wherein each order for a material MA comprises a quantity of the material and a due date by which that quantity is to be provided.

4. The method of Claim 1 wherein each order for a material MA comprises a rate at which the material is to be provided.

5. The method of Claim 1 wherein, for at least one

rank R:

for one material MA1 of said rank R, exploding the order O(MA1) is performed by a processor PR1 of said computer; and

for another material MA2 of the same rank R, exploding the order O(MA2) is performed by another processor PR2 of said computer.

6. The method of Claim 5 wherein said processors PR1 and PR2 explode the respective orders O(MA1) and O(MA2) in parallel.

7. A method of scheduling the production of materials MA1 and MA2, materials MA1 and MA2 being such that neither material is used for producing the other material, by a computer having processors PR1 and PR2 which processors can operate in parallel, said method comprising the steps of:

entering into said computer one or more orders for said material MA1;

entering into said computer one or more orders for said material MA2; and

exploding, by each processor PRi, i=1,2, said orders for the respective material MAi to generate orders for one or more materials needed to produce the respective material MAi.

8. A method for scheduling interrelated events, said method comprising the steps of:

determining a tree relationship among said events such that for each event having one or more child events in the tree, the child events must occur before the parent event;

for one or more of said events, determining one or more entries in a schedule of said events, each entry being a schedule entry for one of said one or more events;

for each rank R in said tree starting from the lowest rank and proceeding in order to the highest rank but one, for each parent event E of the rank R, exploding by a computer one or more entries for the parent event E to generate one or more entries for each child event C(E) of said event E.

9. The method of Claim 8, wherein:

at least one rank R has at least two parent events E1 and E2; and

in said exploding step, one or more entries for E1 are exploded by a processor PR1 of said computer and one or more entries for E2 are exploded by a different processor PR2 of said computer.

10. The method of Claim 9 wherein said processors PR1 and PR2 explode the respective entries for E1 and E2 in parallel.

11. A manufacturing planning system comprising:

one or more processors;

a computer storage;

means for defining in said storage a tree relationship among materials such that each material having, in the tree, one or more child materials is to be produced using said child materials;

means for exploding, by said processors, the orders for all the parent materials of any given rank; and

means for invoking the exploding means for each rank in said tree in sequence from the lowest rank to the highest rank but one.

12. The system of Claim 11 wherein:

said processors are more than one in number; and

said exploding means comprises means for exploding two orders for two respective different

materials by two different processors in parallel, each processor exploding one of said orders.

13. The system of Claim 12 further comprising a control unit for providing instructions to said processors, wherein each processor, when active, executes each instruction provided by said control unit.

14. An event planning system comprising:

a computer storage for storing information representing a tree of events;

means for reading into said computer storage one or more schedule entries for one or more of said events; and

means for exploding schedule entries by one or more processors that can access said storage means, wherein said exploding means, when invoked:

examines parent events of each rank having in said tree a parent event, starting with the lowest rank and proceeding in sequence to the highest rank but one, to determine whether to explode any schedule entry for any parent event;

if the exploding means determines, in examining the parent events of any given rank R, that one or more entries ENT for at least one parent event of rank R are to be exploded, the exploding means explodes said one or more entries ENT before examining parent events of any rank higher than R; and

wherein each entry exploded by said exploding means is either (1) an entry read in by said reading means or (2) an entry generated by said exploding means when the exploding means explodes another entry.

15. The method of Claim 1 further comprising, before said exploding step, the step of determining a rank in

said tree of each material including said materials F and said materials to be used.

16. The method of Claim 1 wherein at least one material MA has in said tree at least two different parent materials P1(MA), P2(MA).

17. The method of Claim 16 wherein a tree level of said at least one material MA as a child of P1(MA) is different from a tree level of said at least one material MA as a child of P2(MA).

18. The method of Claim 8 further comprising, before said exploding step, the step of ranking by said computer each event in said tree.

19. The method of Claim 8 wherein at least one event E has in said tree at least two different parent events P1(E), P2(E).

20. The method of Claim 19 wherein a tree level of said at least one event E as a child of P1(E) is different from a tree level of said at least one event E as a child of P2(E).

21. The system of Claim 11 further comprising means for determining a rank of each said material in said tree.

22. The system of Claim 11 wherein said exploding means and said invoking means can perform their functions when at least one material MA has in said tree at least two different parent materials P1(MA), P2(MA).

23. The system of Claim 22 wherein said exploding means and said invoking means can perform their functions when a tree level of said at least one material MA as a child of P1(MA) is different from a tree level of said at

least one material MA as a child of P2(MA).

24. The system of Claim 14 further comprising means for determining a rank for each event in said tree.

25. The system of Claim 14 wherein said exploding means can perform its function when at least one event E in said tree has at least two parent events P1(E), P2(E).

26. The system of Claim 25 wherein said exploding means can perform its function when a tree level of said at least one event E as a child of P1(E) is different from a tree level of said at least one event E as a child of P2(E).

27. A method for generating schedules for a plurality of units, said method comprising the steps of:

providing for each said unit one or more orders for an output material of a process performed by the unit; and

generating the schedule for each said unit from the respective one or more orders for the unit by a separate computer processor so that said computer processors generate the respective schedules in parallel.

28. The method of Claim 27 wherein said computer processors are part of a computer system that enables communication among said processors.

29. The method of Claim 27 wherein said computer processors are processor elements of an SIMD computer.

30. The method of Claim 27 wherein at least one of said units is a work center.

31. The method of Claim 27 wherein at least one of

said units is an outside supplier.

32. A system for generating schedules for a plurality of units, said system comprising:

a computer system comprising a plurality of computer processors; and

means for operating said computer processors in parallel to generate a schedule for each said unit by a separate one of said computer processors.

33. The system of Claim 32 wherein said computer system comprises an SIMD computer.

34. A method for performing checking of schedules of a plurality of units for overload, said method comprising the steps of: -

providing a plurality of computer processors; and

checking for overload the schedule of each said unit by a separate one of said computer processors so that all said computer processors perform the respective checking in parallel.

35. The method of Claim 34 wherein said computer processors are part of a computer system that enables communication among said processors.

36. The method of Claim 34 wherein said computer processors are processor elements of an SIMD computer.

37. A system for performing checking of schedules of a plurality of units for overload, said system comprising:

a computer system comprising a plurality of computer processors; and

means for operating said computer processors in parallel so as to check each said schedule for overload by a separate one of said computer

processors.

38. The system of Claim 37 wherein said computer system comprises an SIMD computer.

39. A method of transferring information among a plurality of processors in a system in which more than one processors can send information to one or more other processors parallel, said method comprising the steps of:

marking each processor that has information to send to another processor;

while at least one processor is marked, performing the steps of:

determining a set of one or more marked processors that can send information in parallel with each other;

unmarking the processors of said set; and

sending information by the processors of said set in parallel.

40. The method of Claim 39 wherein said marking step comprises the step of setting in parallel by each said processor a respective variable to a predetermined value.

41. The method of Claim 39 wherein said processors are processor elements of an SIMD computer.

42. A computer system comprising:

a plurality of processors such that more than one processors can send information to one or more other processors in parallel;

means for marking each processor that has information to send to another processor;

means for determining a set of one or more marked processors that can send information in parallel with each other; and

means for unmarking the processors of said set.

43. The system of Claim 42 wherein said processors are processor elements of an SIMD computer.

44. A method for simulating a manufacturing system, said method comprising the steps of:

determining a tree relationship among materials such that each material which has in the tree one or more child materials is to be produced using its child materials;

determining for said materials by a computer initial inventories and orders to outside suppliers;

for each rank R in said tree starting with the highest rank but one and proceeding in sequence to the lowest rank, determining by said computer from orders, and inventories, for one or more materials of one or more ranks higher than R and from inventories for one more materials of the rank R orders that can be satisfied for one or more materials of the rank R.

45. The method of Claim 44 further comprising, before said step of determining orders, the step of determining by said computer a rank of each said material in said tree.

46. The method of Claim 44 wherein said step of determining orders comprises, for at least one rank R, the step of determining orders by at least two different processors operating in parallel, each processor determining orders for a separate material of the rank R.

47. The method of Claim 44 wherein at least one material MA has in said tree at least two different parent materials P1(MA), P2(MA).

48. The method of Claim 47 wherein a tree level of said at least one material MA as a child of P1(MA) is different from a tree level of said at least one material

MA as a child of P2(MA).

49. A computer system for simulating manufacturing, said system comprising:

one or more processors;

a computer storage;

means for defining in said storage a tree relationship among materials such that each material having in the tree one or more child materials is to be produced using said child materials;

means for determining for said materials by said computer initial inventories and orders to outside suppliers;

means for determining by said one or more processors, for each rank R in said tree starting with the highest rank but one and proceeding in sequence to the lowest rank, from orders, and inventories, of one or more materials of one or more ranks higher than R and from inventories for one or more materials of the rank R orders that can be satisfied for one or more materials of the rank R.

50. The system of Claim 49 further comprising means for determining a rank of each said material in said tree.

51. The system of Claim 49 comprising more than one processors,

wherein said means for determining orders that can be satisfied comprises means for operating at least two of said processors in parallel so that, for at least one rank R, each processor determines said orders that can be satisfied for a separate material of the rank R.

52. The system of Claim 49 wherein said means for determining orders that can be satisfied can perform its function when at least one material MA has in said tree at

least two parent materials P1(MA), P2(MA).

53. The system of Claim 52 wherein said means for determining orders that can be satisfied can perform its function when a tree level of said at least one material MA as a child of P1(MA) is different from a tree level of said at least one material MA as a child of P2(MA).

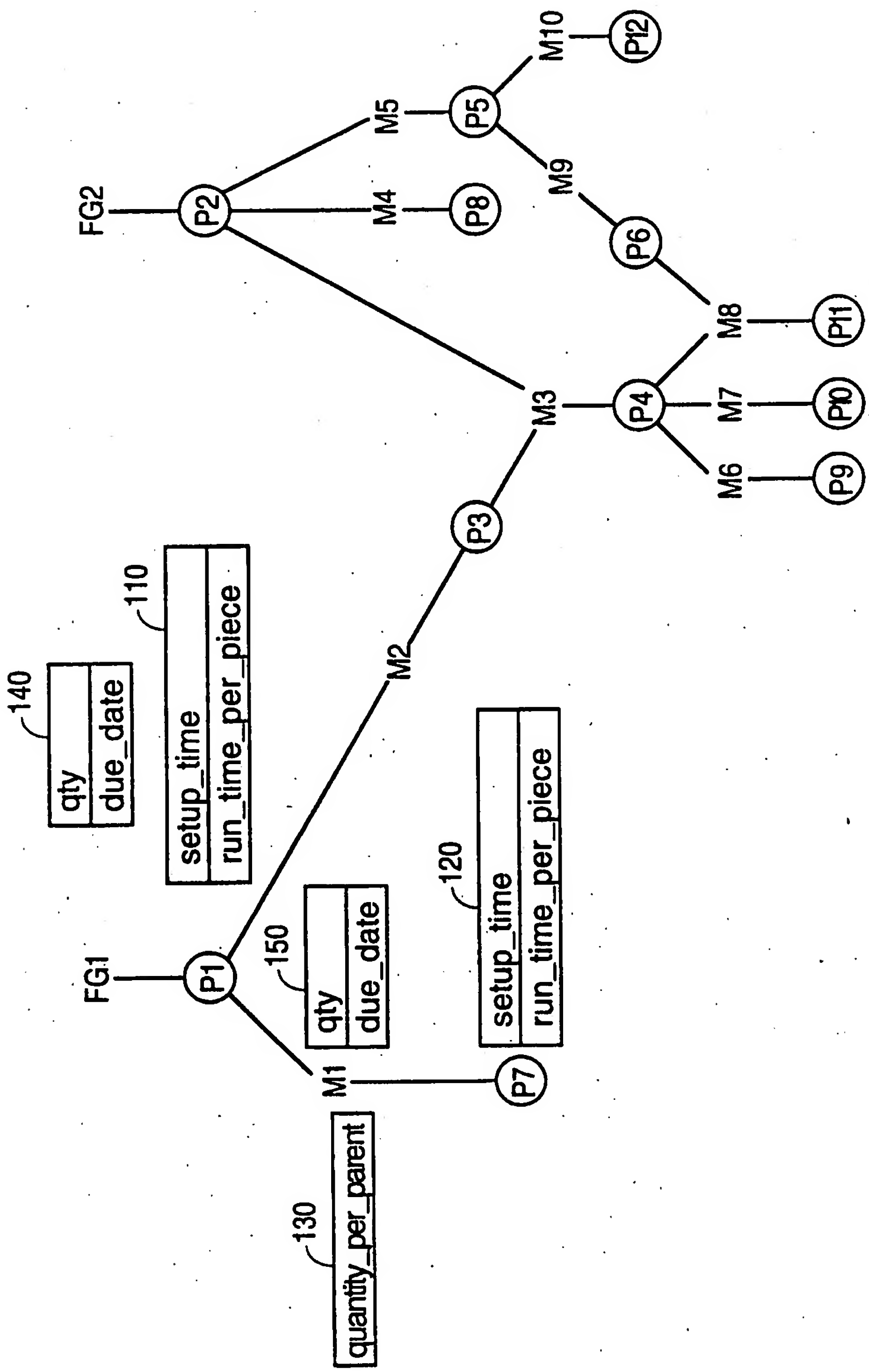

FG1
FG2
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
140
qty
due_date
110
setup_time
run_time_per_piece
150
qty
due_date
120
setup_time
run_time_per_piece
130
quantity_per_parent

FIG. 1

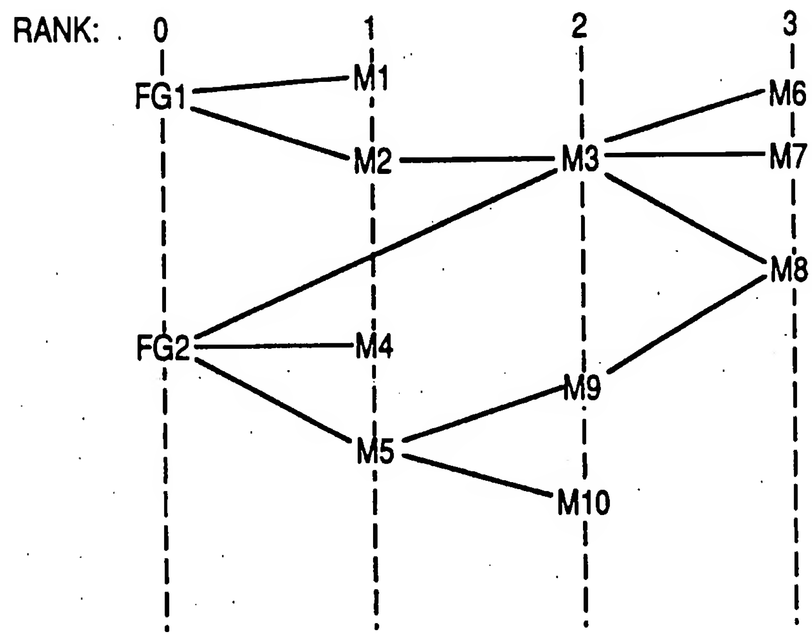


FIG. 2

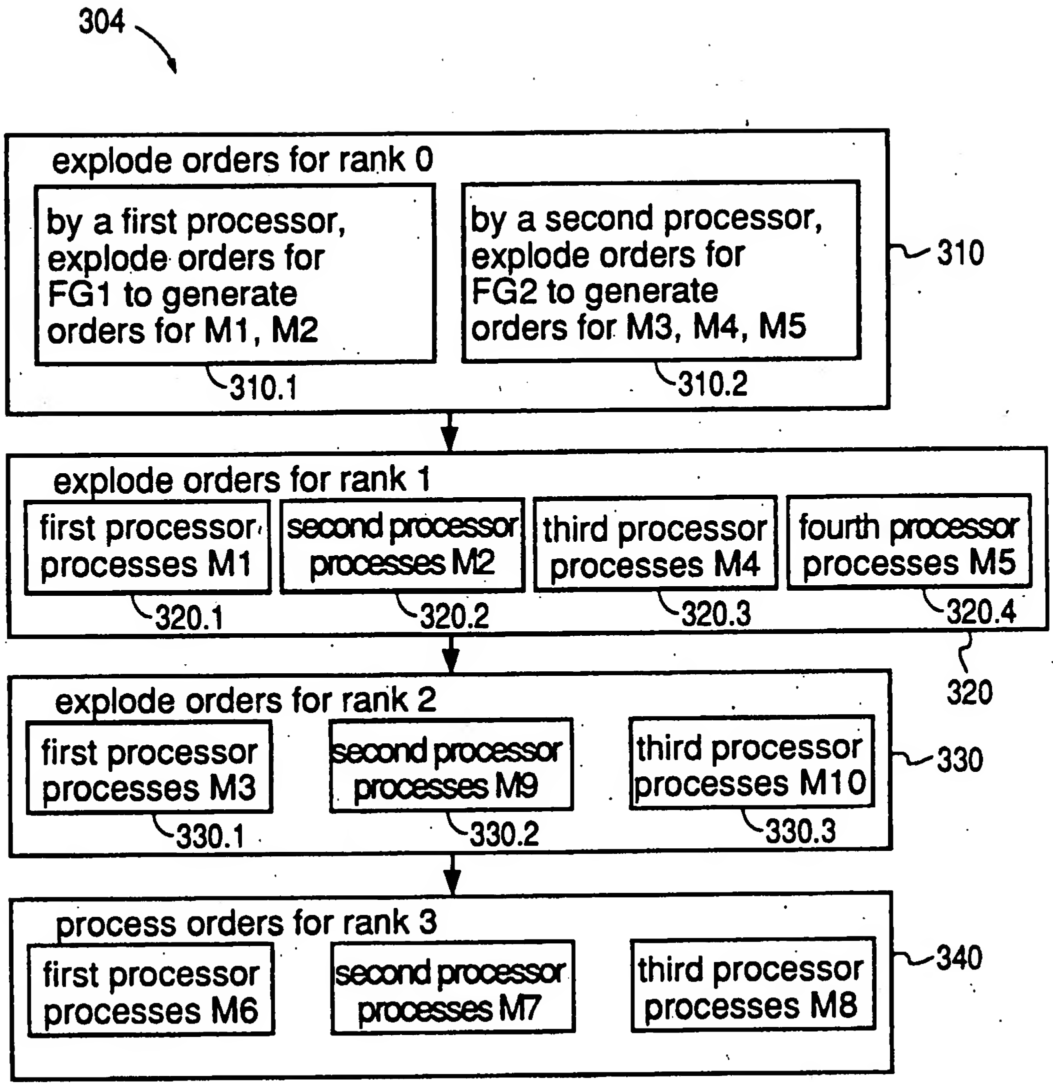

304
explode orders for rank 0
by a first processor, explode orders for FG1 to generate orders for M1, M2
310.1
by a second processor, explode orders for FG2 to generate orders for M3, M4, M5
310.2
310
explode orders for rank 1
first processor processes M1
320.1
second processor processes M2
320.2
third processor processes M4
320.3
fourth processor processes M5
320.4
320
explode orders for rank 2
first processor processes M3
330.1
second processor processes M9
330.2
third processor processes M10
330.3
330
process orders for rank 3
first processor processes M6
second processor processes M7
third processor processes M8
340

FIG. 3

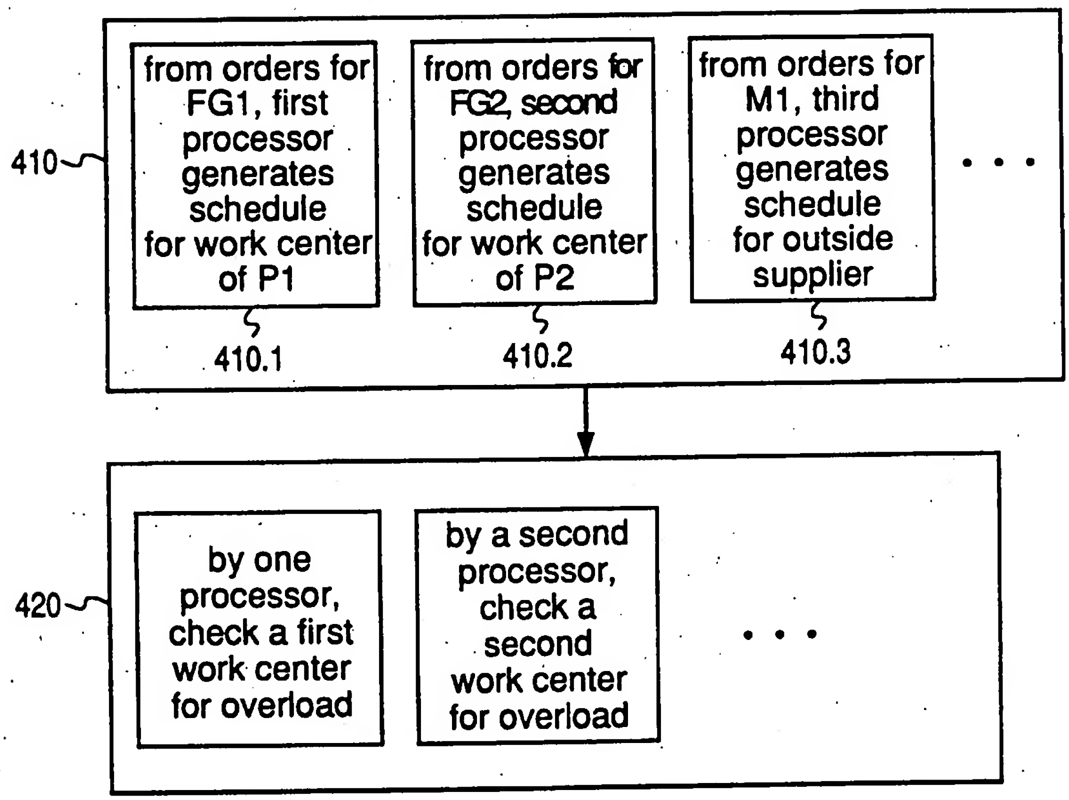

410
from orders for FG1, first processor generates schedule for work center of P1
410.1
from orders for FG2, second processor generates schedule for work center of P2
410.2
from orders for M1, third processor generates schedule for outside supplier
410.3
420
by one processor, check a first work center for overload
by a second processor, check a second work center for overload

FIG. 4

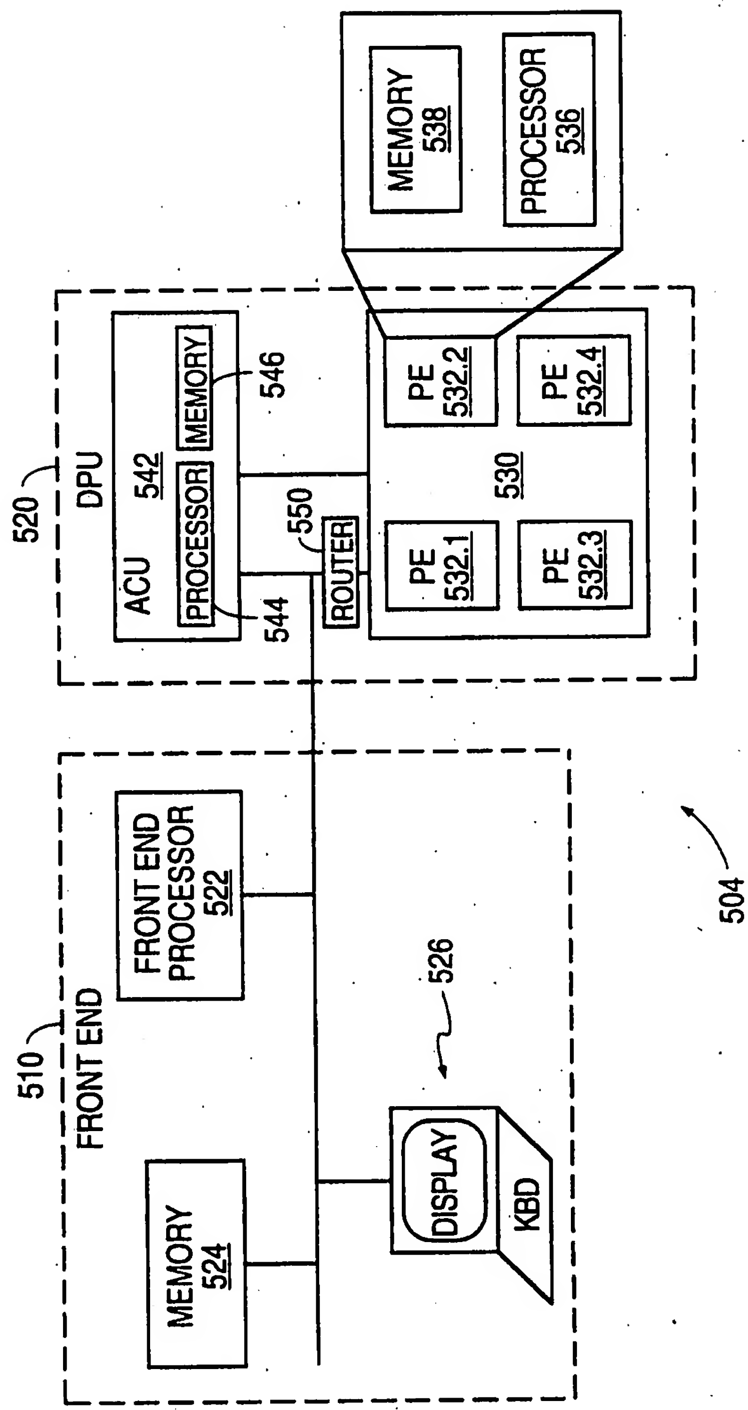

MEMORY
538
PROCESSOR
536
520
DPU
ACU
542
PROCESSOR
MEMORY
546
544
550
ROUTER
PE
532.1
PE
532.2
PE
532.3
PE
532.4
530
510
FRONT END
FRONT END
PROCESSOR
522
MEMORY
524
526
DISPLAY
KBD
504

FIG. 5

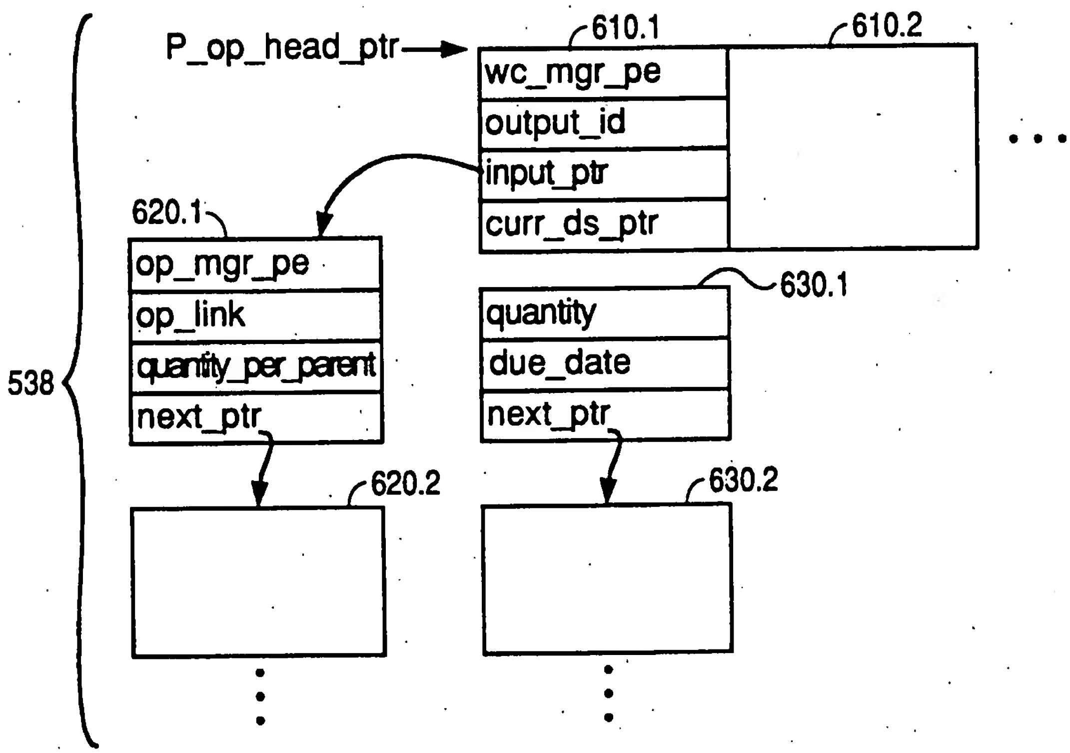

P_op_head_ptr
610.1
610.2
wc_mgr_pe
output_id
input_ptr
curr_ds_ptr
620.1
op_mgr_pe
op_link
quantity_per_parent
next_ptr
630.1
quantity
due_date
next_ptr
538
620.2
630.2

FIG. 6A

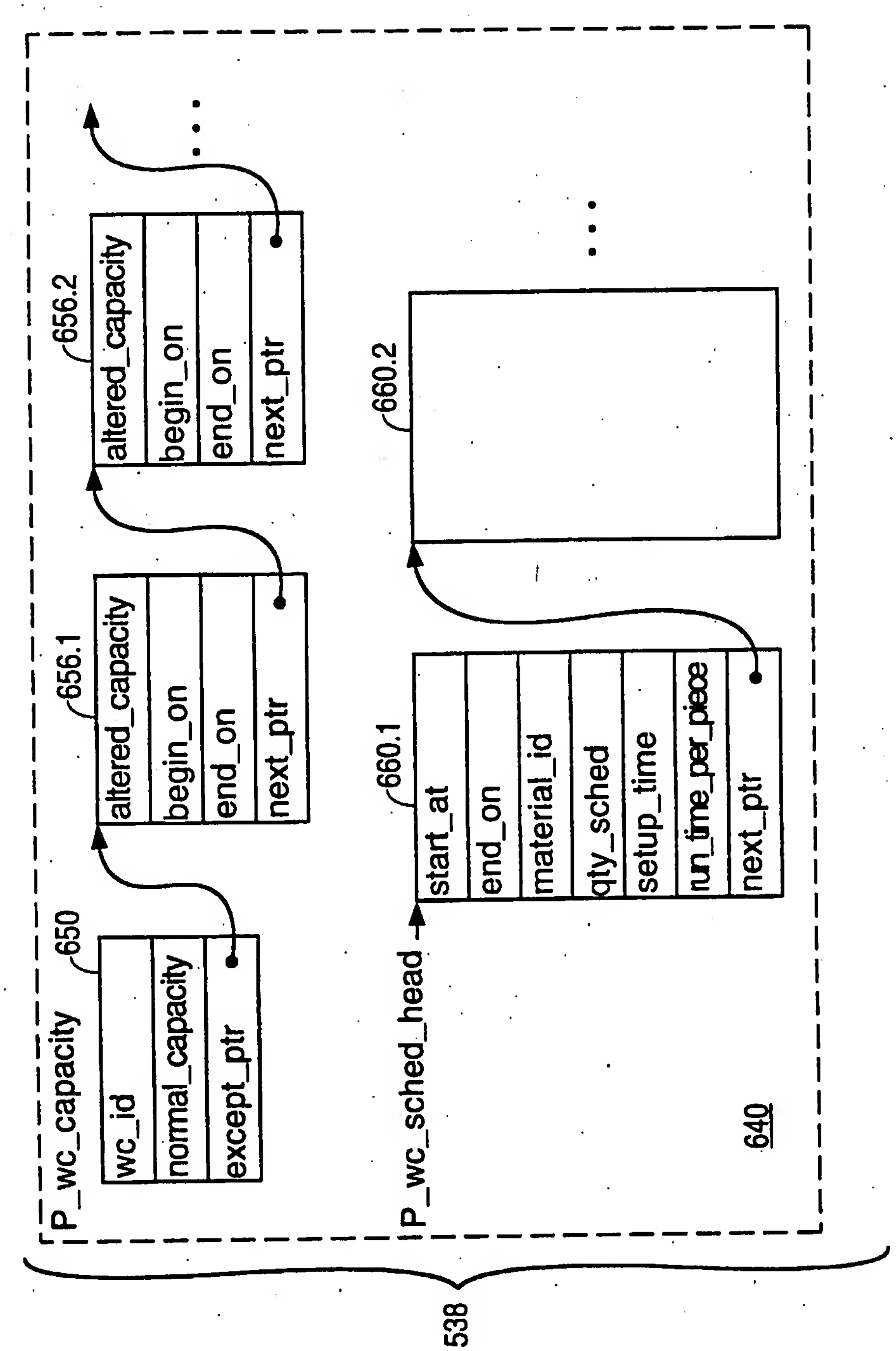

P_wc_capacity
650
wc_id
normal_capacity
except_ptr
656.1
altered_capacity
begin_on
end_on
next_ptr
656.2
altered_capacity
begin_on
end_on
next_ptr
P_wc_sched_head
660.1
start_at
end_on
material_id
qty_sched
setup_time
run_time_per_piece
next_ptr
660.2
640
538

FIG. 6B

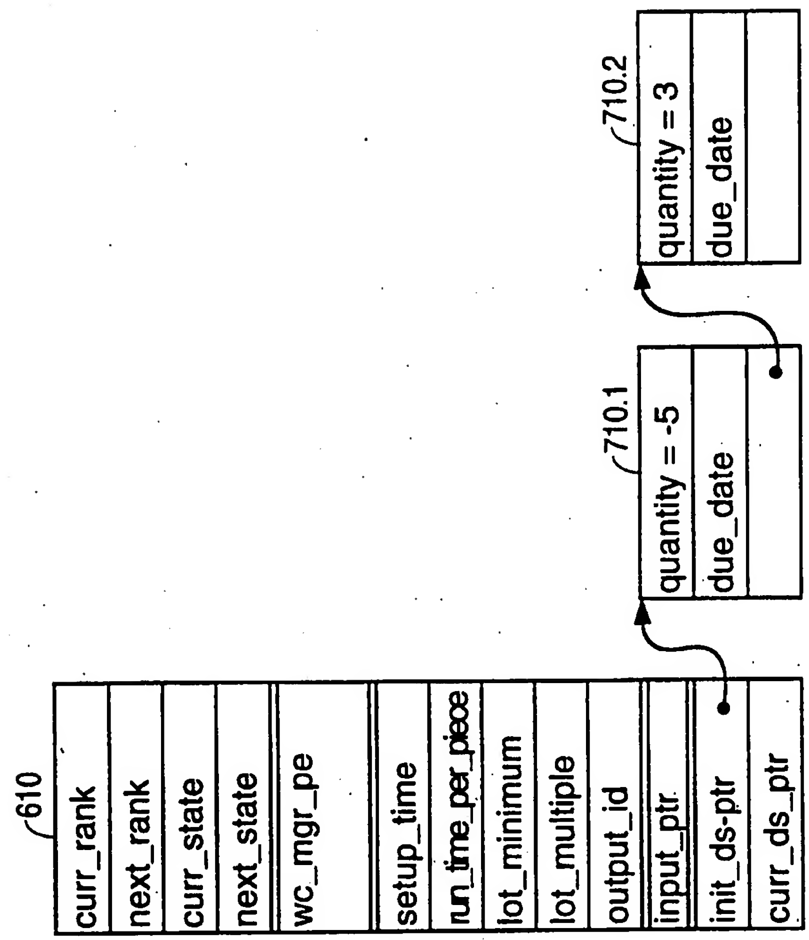

610
curr_rank
next_rank
curr_state
next_state
wc_mgr_pe
setup_time
run_time_per_piece
lot_minimum
lot_multiple
output_id
input_ptr
init_ds-ptr
curr_ds_ptr
710.1
quantity = -5
due_date
710.2
quantity = 3
due_date

FIG. 7

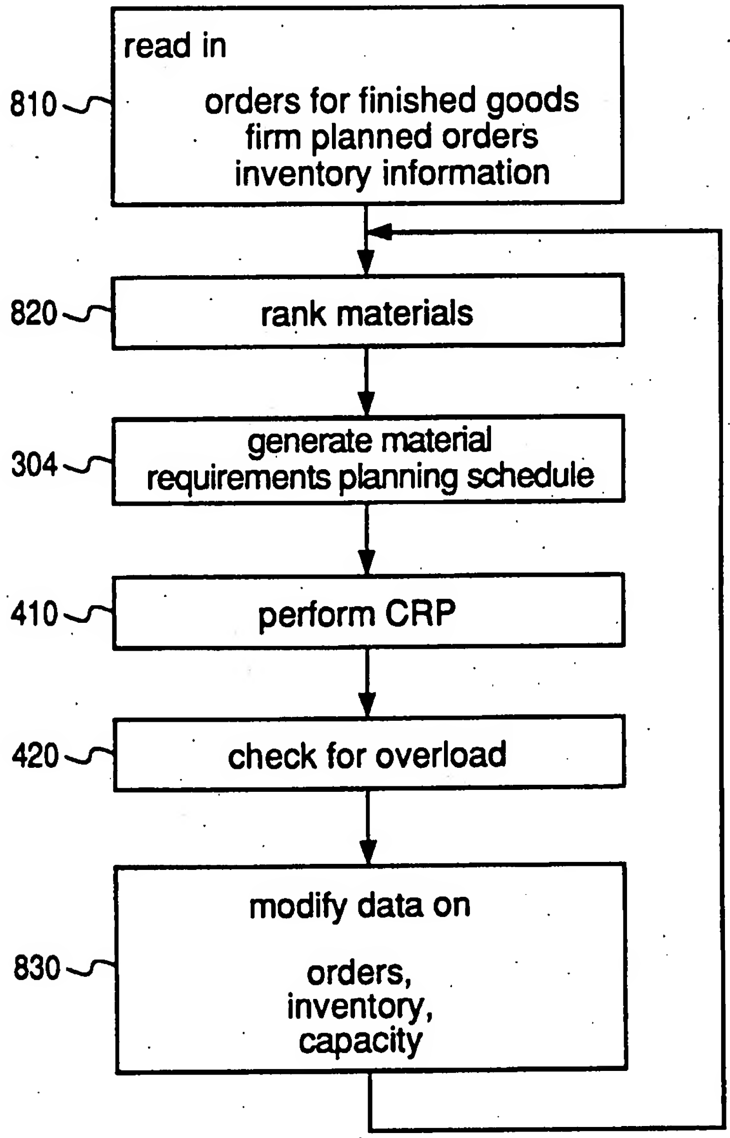

read in
orders for finished goods
firm planned orders
inventory information
810
rank materials
820
generate material requirements planning schedule
304
perform CRP
410
check for overload
420
modify data on
orders,
inventory,
capacity
830

FIG. 8

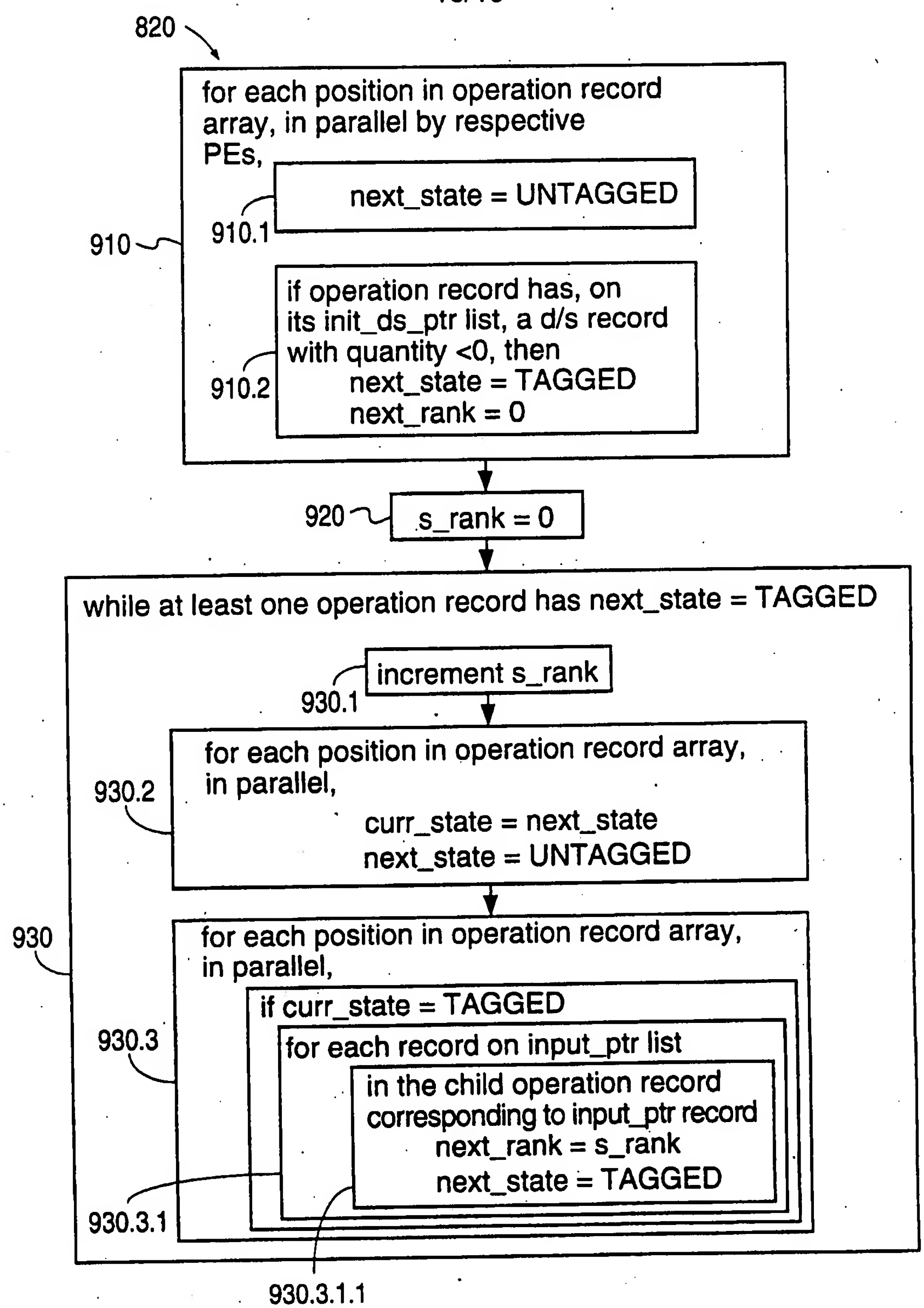

820
for each position in operation record array, in parallel by respective PEs,
next_state = UNTAGGED
910.1
910
if operation record has, on its init_ds_ptr list, a d/s record with quantity <0, then
next_state = TAGGED
next_rank = 0
910.2
920
s_rank = 0
while at least one operation record has next_state = TAGGED
increment s_rank
930.1
for each position in operation record array, in parallel,
curr_state = next_state
next_state = UNTAGGED
930.2
930
for each position in operation record array, in parallel,
if curr_state = TAGGED
for each record on input_ptr list
in the child operation record corresponding to input_ptr record
next_rank = s_rank
next_state = TAGGED
930.3
930.3.1
930.3.1.1

FIG. 9

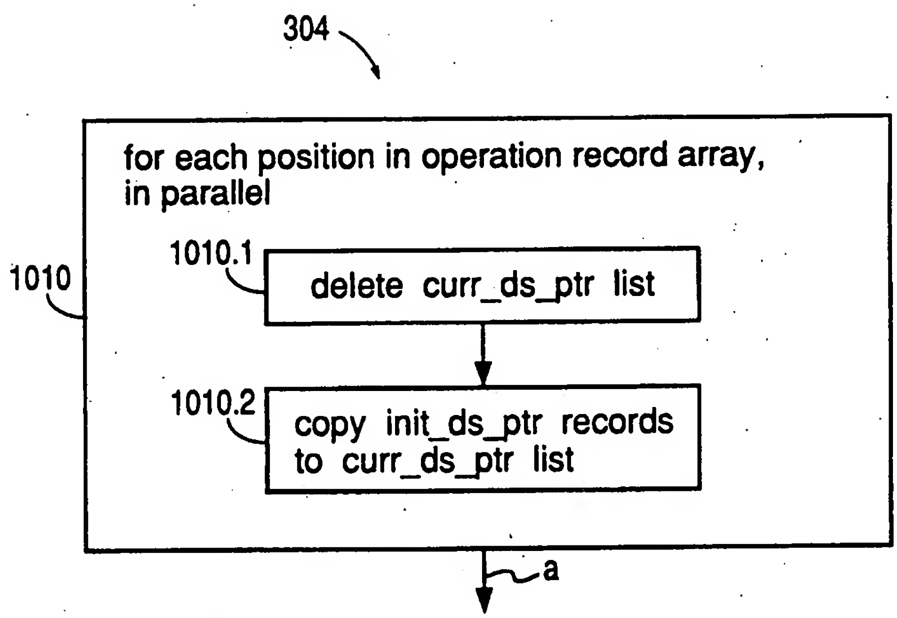

304
for each position in operation record array, in parallel
1010
1010.1
delete curr_ds_ptr list
1010.2
copy init_ds_ptr records to curr_ds_ptr list
a

FIG. 10A

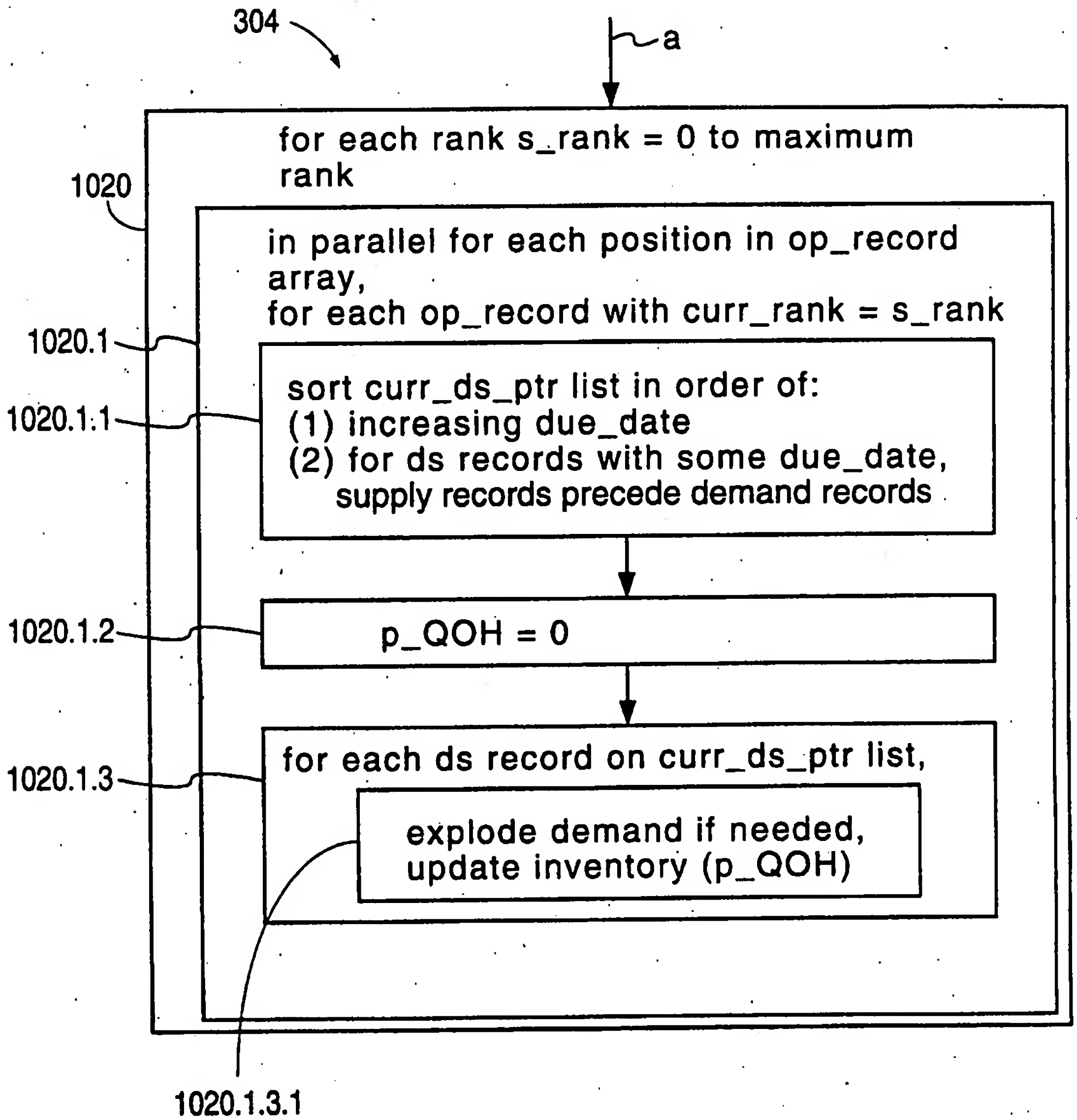

304
a
1020
for each rank s_rank = 0 to maximum rank
1020.1
in parallel for each position in op_record array,
for each op_record with curr_rank = s_rank
1020.1.1
sort curr_ds_ptr list in order of:
(1) increasing due_date
(2) for ds records with some due_date, supply records precede demand records
1020.1.2
p_QOH = 0
1020.1.3
for each ds record on curr_ds_ptr list,
explode demand if needed, update inventory (p_QOH)
1020.1.3.1

FIG. 10B

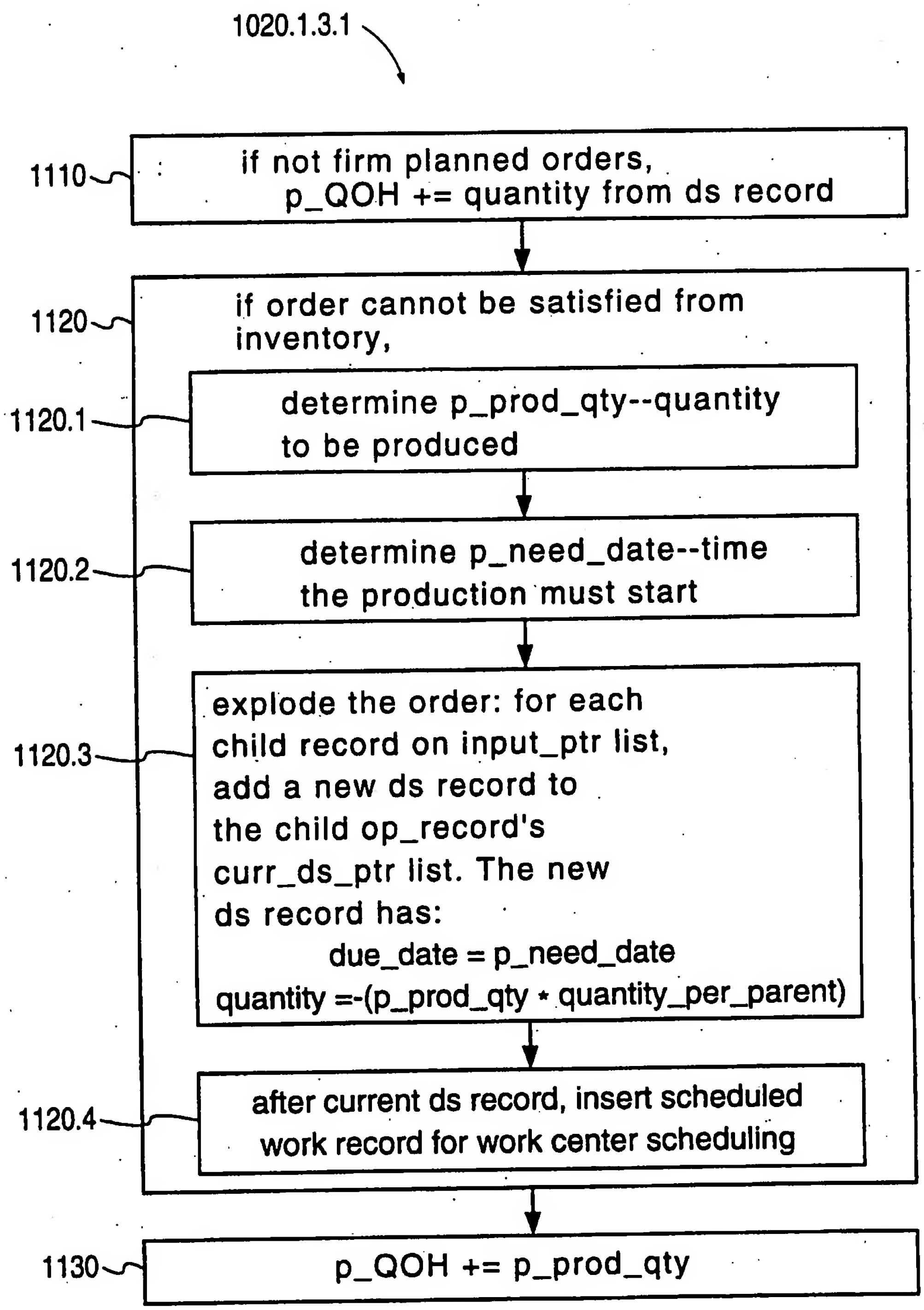

1020.1.3.1
1110
if not firm planned orders,
p_QOH += quantity from ds record
1120
if order cannot be satisfied from inventory,
1120.1
determine p_prod_qty--quantity to be produced
1120.2
determine p_need_date--time the production must start
1120.3
explode the order: for each child record on input_ptr list, add a new ds record to the child op_record's curr_ds_ptr list. The new ds record has:
due_date = p_need_date
quantity =-(p_prod_qty * quantity_per_parent)
1120.4
after current ds record, insert scheduled work record for work center scheduling
1130
p_QOH += p_prod_qty

FIG. 11

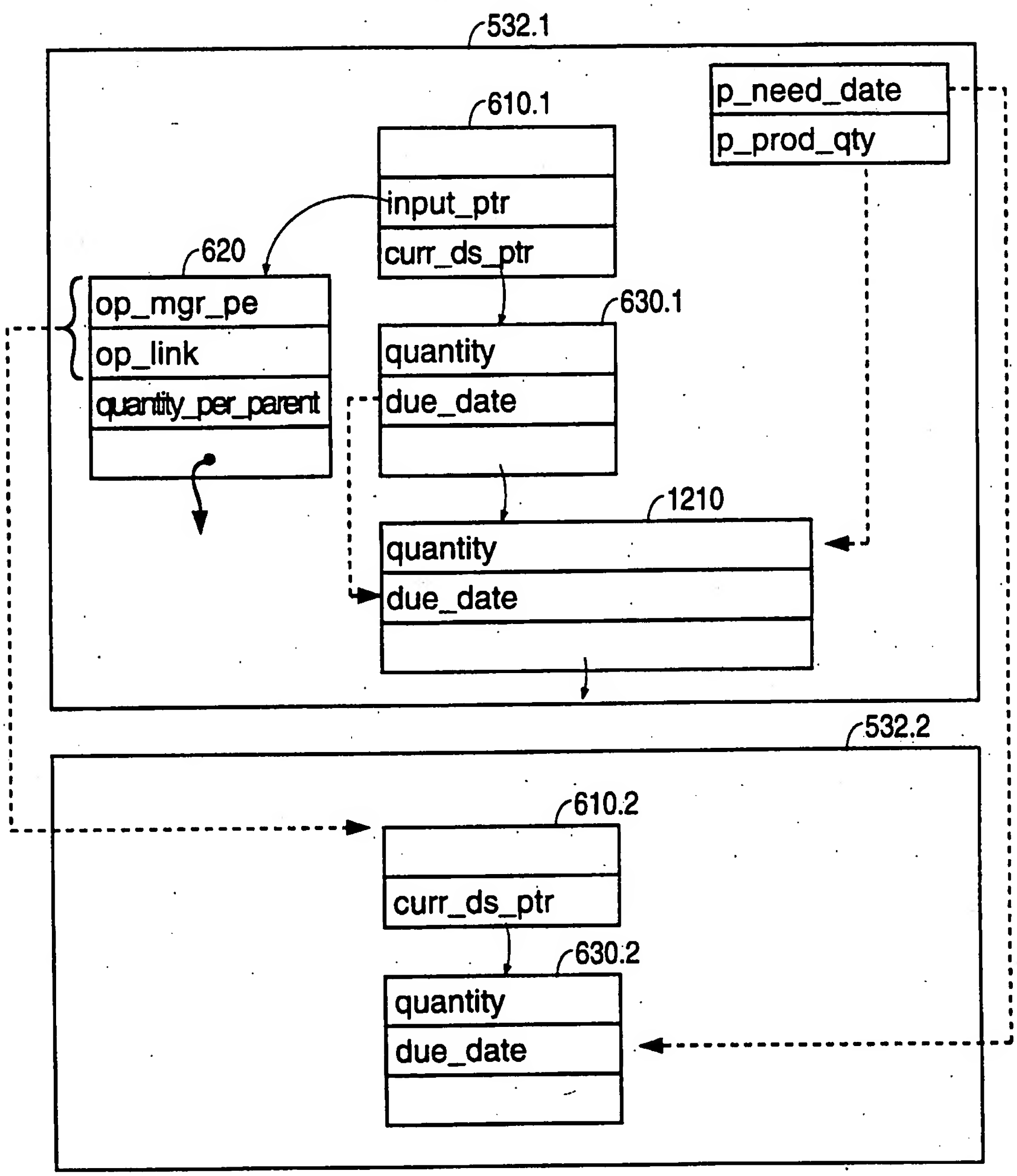

532.1
610.1
p_need_date
p_prod_qty
input_ptr
curr_ds_ptr
620
op_mgr_pe
op_link
quantity_per_parent
630.1
quantity
due_date
1210
quantity
due_date
532.2
610.2
curr_ds_ptr
630.2
quantity
due_date

FIG. 12

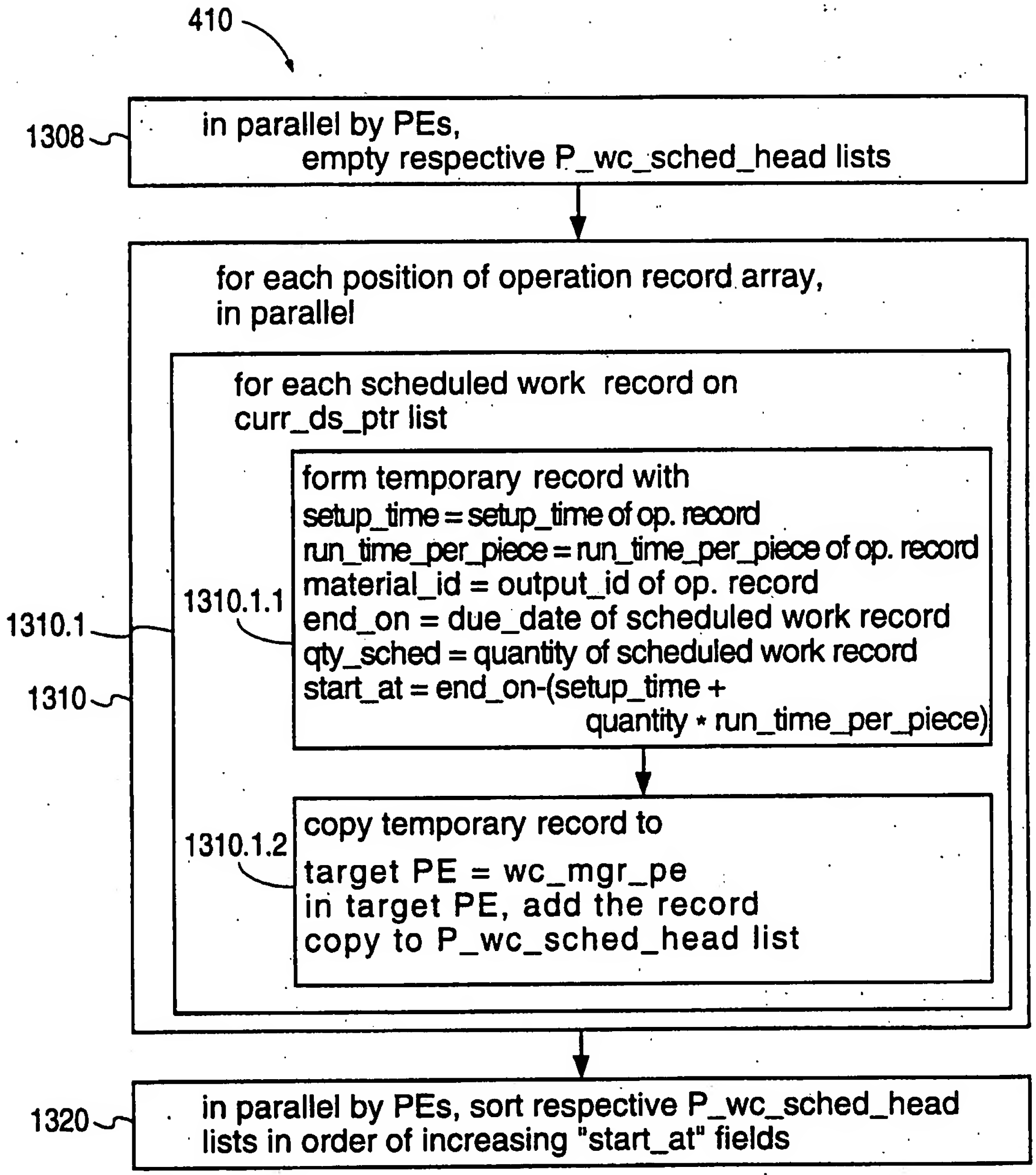

410
1308
in parallel by PEs,
empty respective P_wc_sched_head lists
for each position of operation record array,
in parallel
for each scheduled work record on
curr_ds_ptr list
1310.1
1310
1310.1.1
form temporary record with
setup_time = setup_time of op. record
run_time_per_piece = run_time_per_piece of op. record
material_id = output_id of op. record
end_on = due_date of scheduled work record
qty_sched = quantity of scheduled work record
start_at = end_on-(setup_time +
quantity * run_time_per_piece)
1310.1.2
copy temporary record to
target PE = wc_mgr_pe
in target PE, add the record
copy to P_wc_sched_head list
1320
in parallel by PEs, sort respective P_wc_sched_head
lists in order of increasing "start_at" fields

FIG. 13

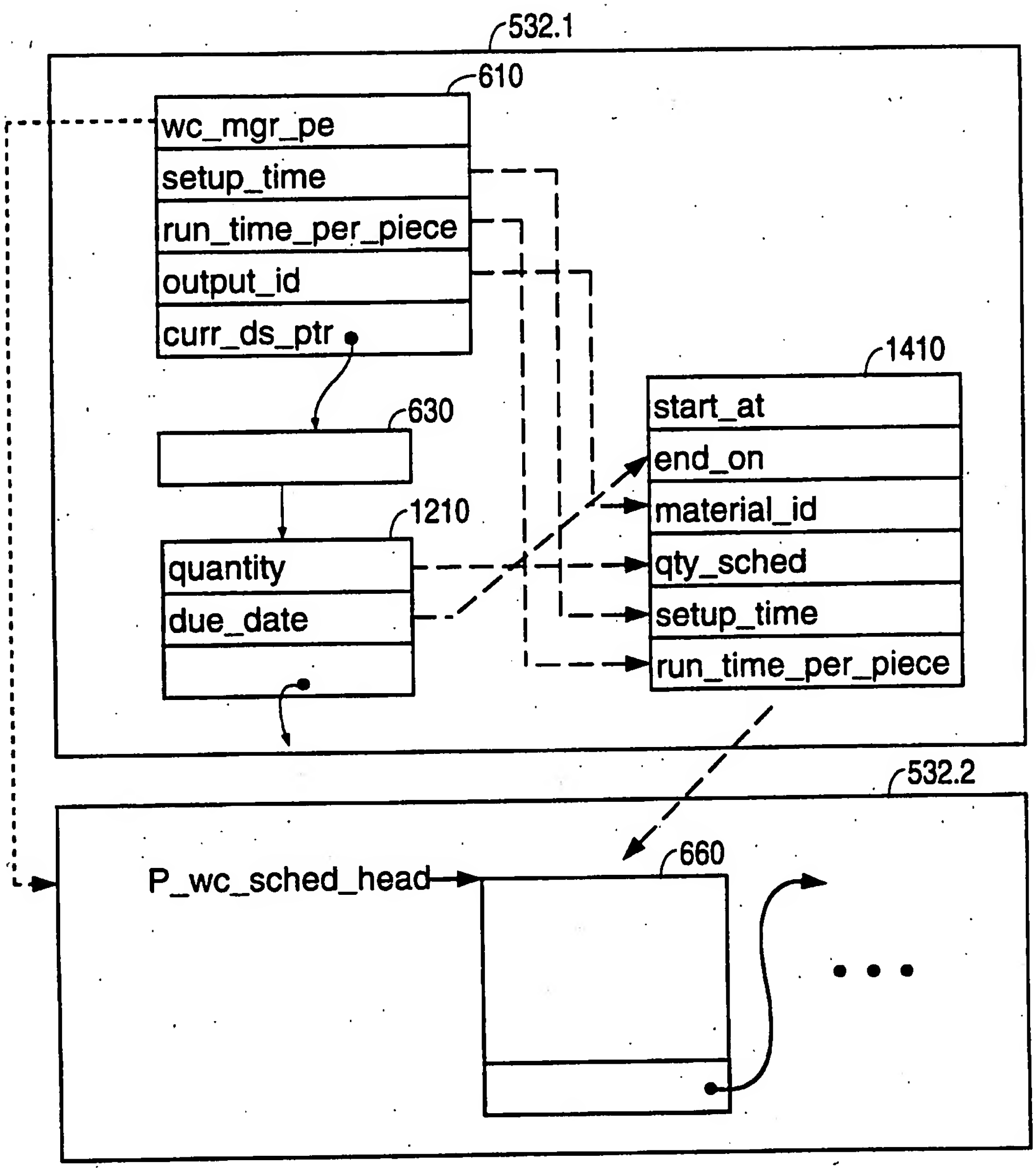

532.1
610
wc_mgr_pe
setup_time
run_time_per_piece
output_id
curr_ds_ptr
630
1210
quantity
due_date
1410
start_at
end_on
material_id
qty_sched
setup_time
run_time_per_piece
532.2
660
P_wc_sched_head

FIG. 14

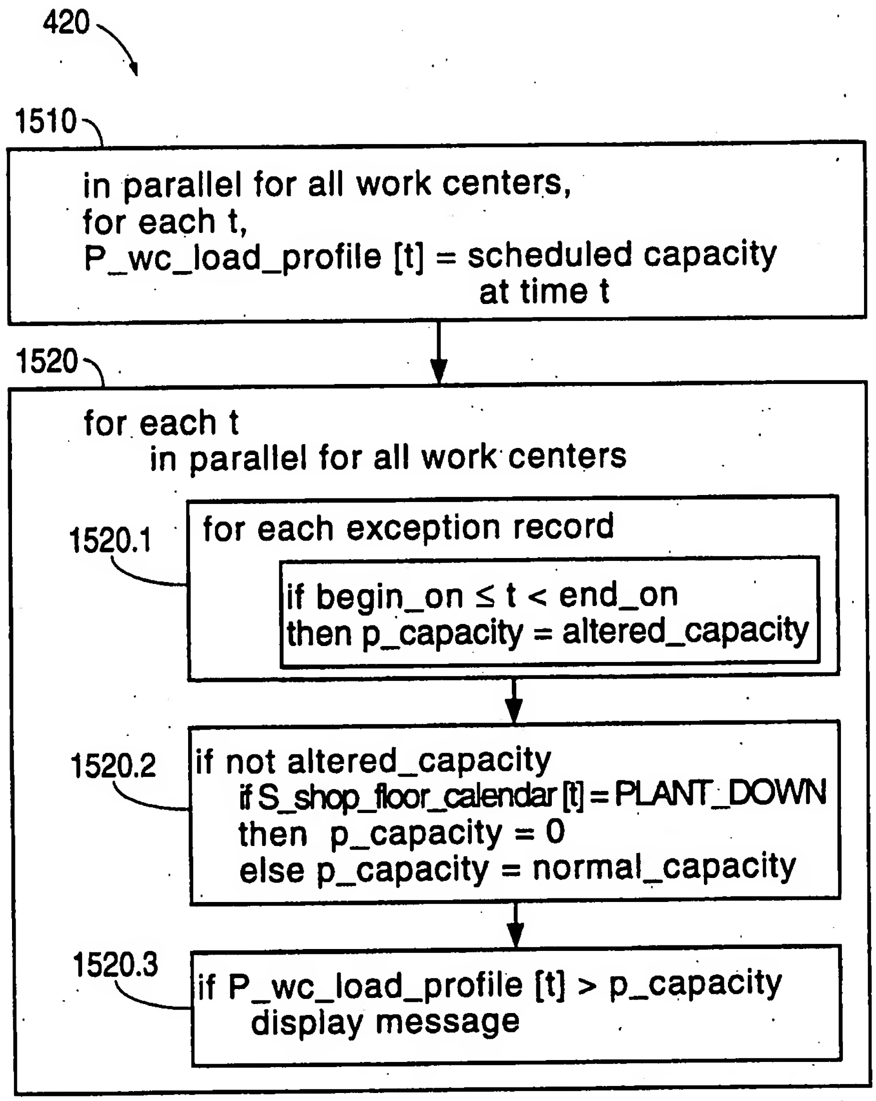

420
1510
in parallel for all work centers,
for each t,
P_wc_load_profile [t] = scheduled capacity at time t
1520
for each t
in parallel for all work centers
1520.1
for each exception record
if begin_on ≤ t < end_on
then p_capacity = altered_capacity
1520.2
if not altered_capacity
if S_shop_floor_calendar [t] = PLANT_DOWN
then p_capacity = 0
else p_capacity = normal_capacity
1520.3
if P_wc_load_profile [t] > p_capacity
display message

FIG. 15

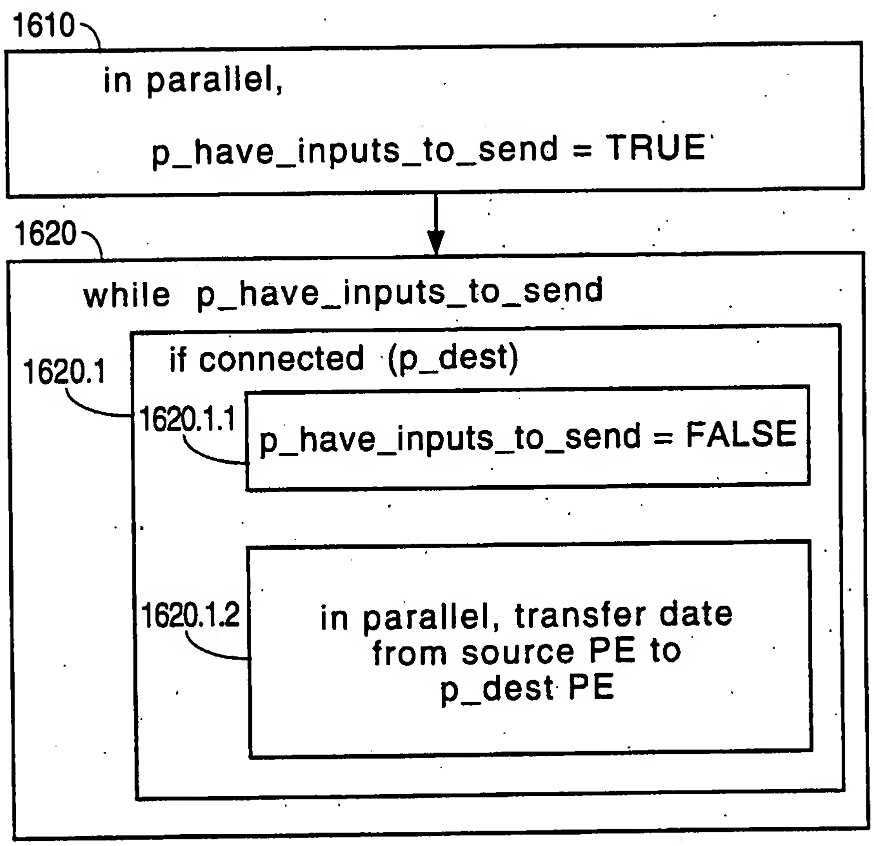

1610
in parallel,
p_have_inputs_to_send = TRUE
1620
while p_have_inputs_to_send
1620.1
if connected (p_dest)
1620.1.1
p_have_inputs_to_send = FALSE
1620.1.2
in parallel, transfer date
from source PE to
p_dest PE

FIG. 16

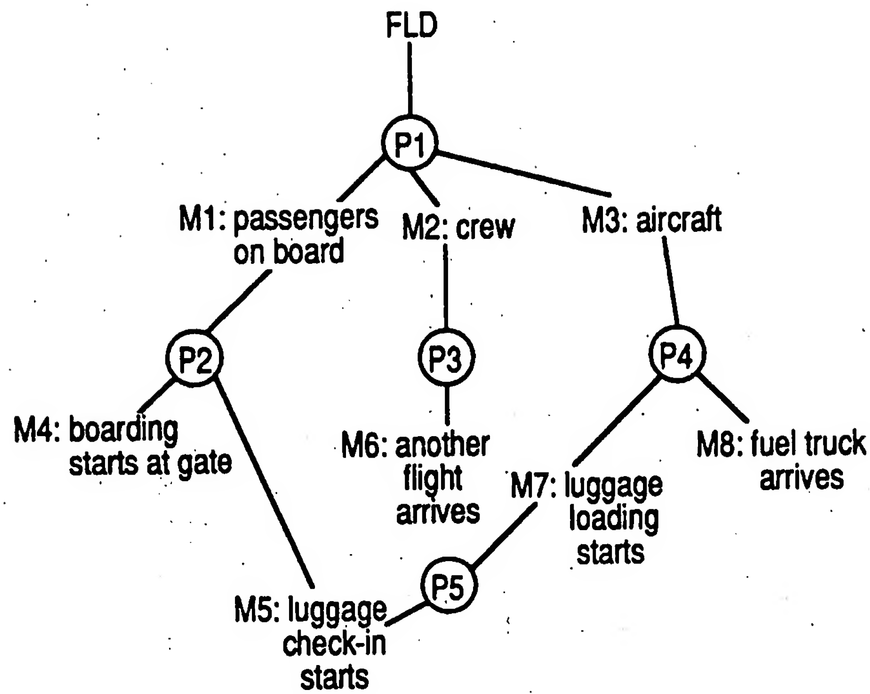

FLD
P1
M1: passengers on board
M2: crew
M3: aircraft
P2
P3
P4
M4: boarding starts at gate
M6: another flight arrives
M7: luggage loading starts
M8: fuel truck arrives
P5
M5: luggage check-in starts

FIG. 17